Lecture 5

Derivatives and edges

Administrative

A1 is out
- It is graded
- Due **Tue, Apr 16**

A2 will be out this weekend

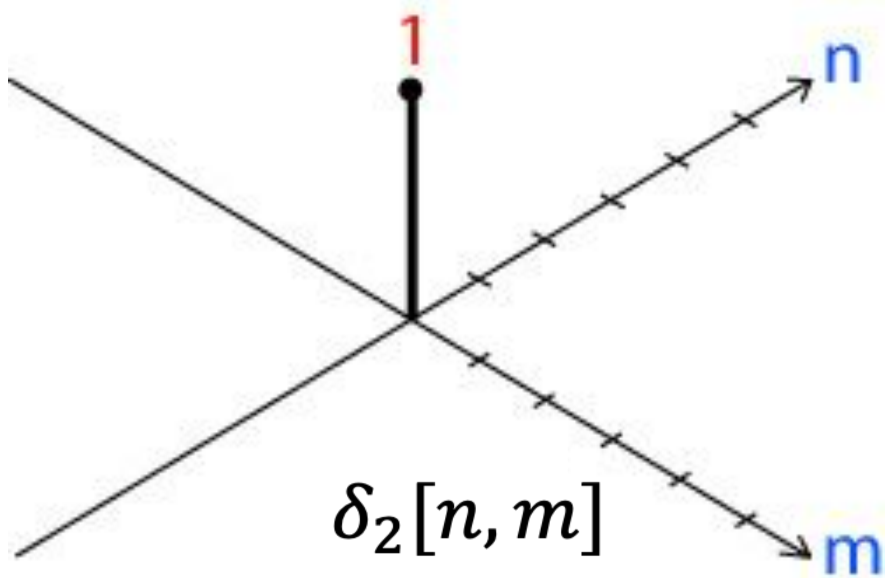
Administrative

Recitation this friday:
More linear algebra recap

by Mahtab

So far: 2D impulse function

- A special function
- 1 at the origin [0,0].
- 0 everywhere else



So far: We get the impulse response when we pass an impulse function through a LSI system

- The moving average filter equation again: $g[n,m] = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-k, m-l]$

$$\delta_2[n,m] \xrightarrow{S} h[n,m]$$

Pass in an impulse function — Record its response

$$h[n,m] = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

So far: write down f as a sum of impulses

Let's say our input f is a 3x3 image:

f[0,0]	f[0,1]	f[1,1]
f[1,0]	f[1,1]	f[1,2]
f[2,0]	f[2,1]	f[2,2]

=

f[0,0]	0	0
0	0	0
0	0	0

+

0	f[0,1]	0
0	0	0
0	0	0

+ … +

0	0	0
0	0	0
0	0	f[2,2]

= f[0,0] ×

1	0	0
0	0	0
0	0	0

+ f[0,1] ×

0	1	0
0	0	0
0	0	0

+ … + f[2,2] ×

0	0	0
0	0	0
0	0	1

$$= f[0,0] \cdot \delta_2[n,m] + f[0,1] \cdot \delta_2[n,m-1] + \ldots + f[2,2] \cdot \delta_2[n-2,m-2]$$

So far: We derived convolutions

- An LSI system is completely specified by its impulse response.
    - For any input f, we can compute the output g in terms of the impulse response h.

Discrete Convolution

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k, m-l]$$

So far: We created a sharpening system by combining filters

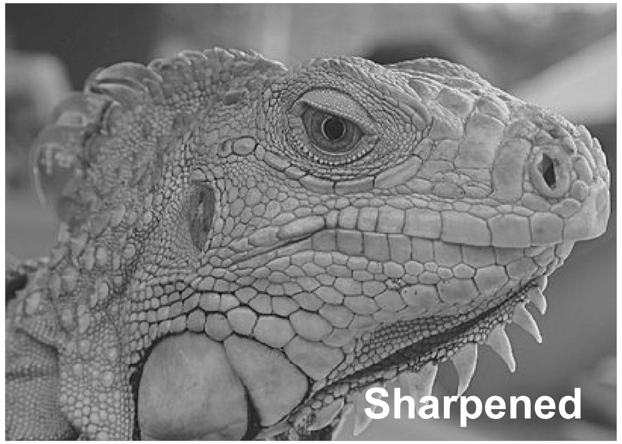
–

=

Let's add it back to get a sharpening system:

+

=



(Cross) correlation – symbol: $**$

Cross correlation of two 2D signals f[n,m] and h[n,m]

$$f[n,m] ** h[n,m] = \sum_{k} \sum_{l} f[k,l] h[n+k, m+l]$$

Equivalent to a convolution without the flip

What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector

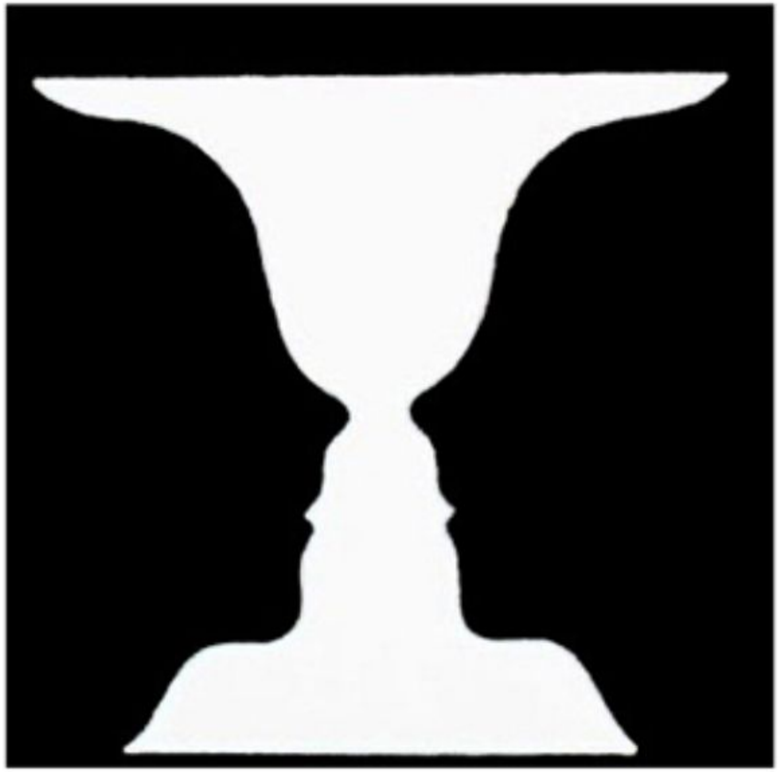
Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 8

What we will learn today

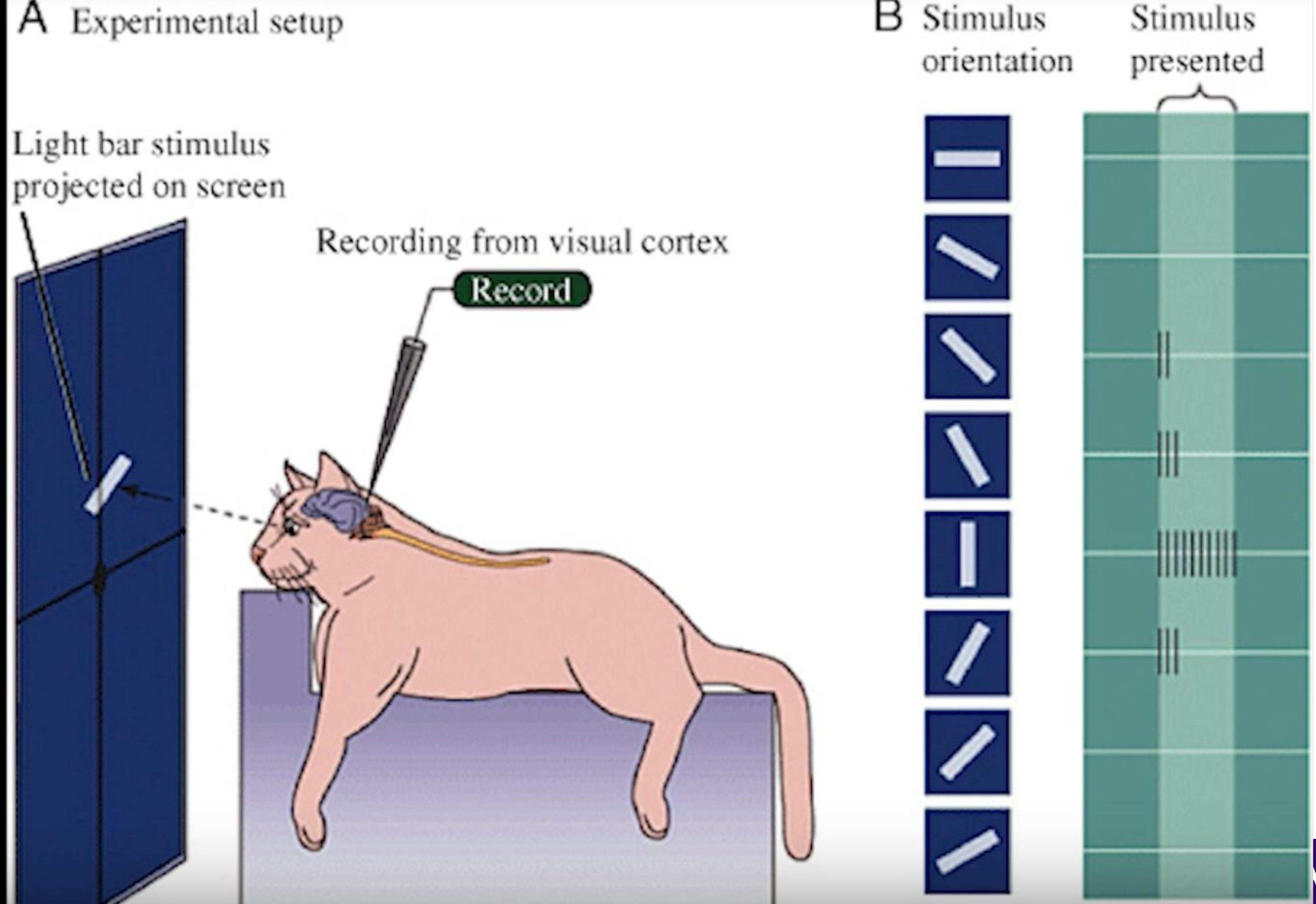
- Edge detection
- Image Gradients
- A simple edge detector

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 8

Q. What do you see?

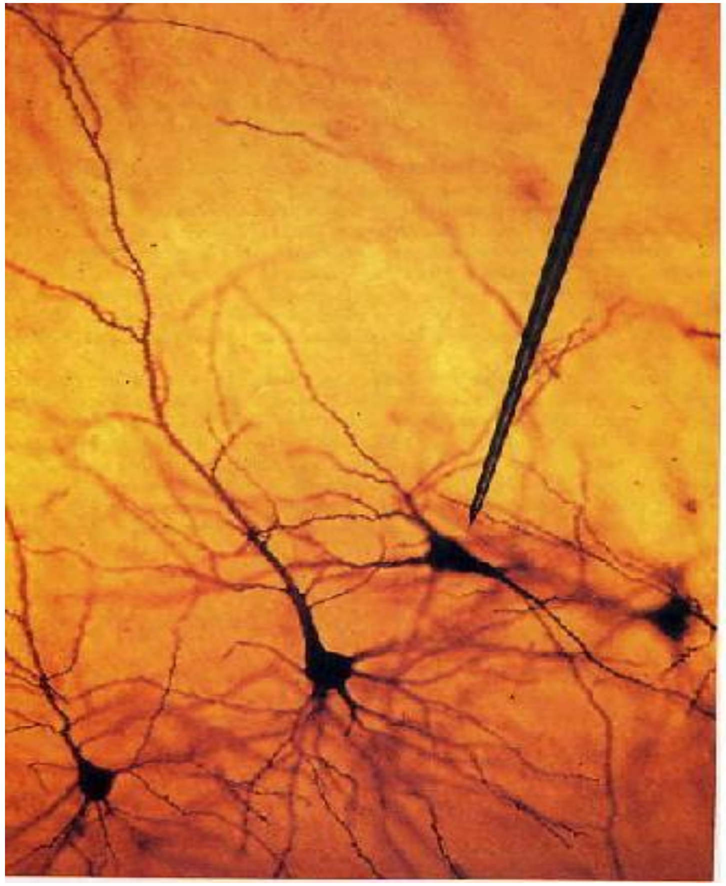


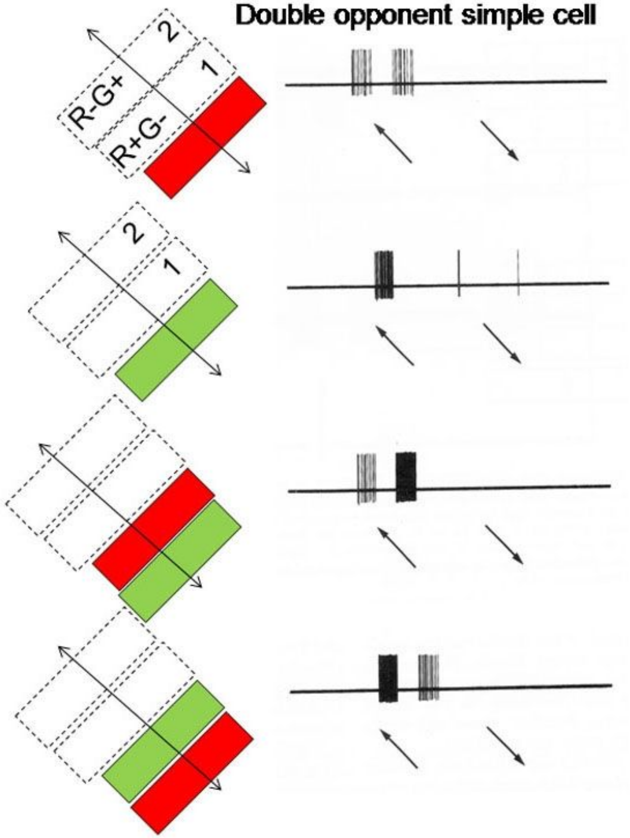
(A) Cave painting at Chauvet, France, about 30,000 B.C.;
(B) Aerial photograph of the picture of a monkey as part of the Nazca Lines geoglyphs, Peru, about 700 – 200 B.C.;
(C) Shen Zhou (1427-1509 A.D.): Poet on a mountain top, ink on paper, China;
(D) Line drawing by 7-year old I. Lleras (2010 A.D.).



Hubel & Wiesel, 1960s

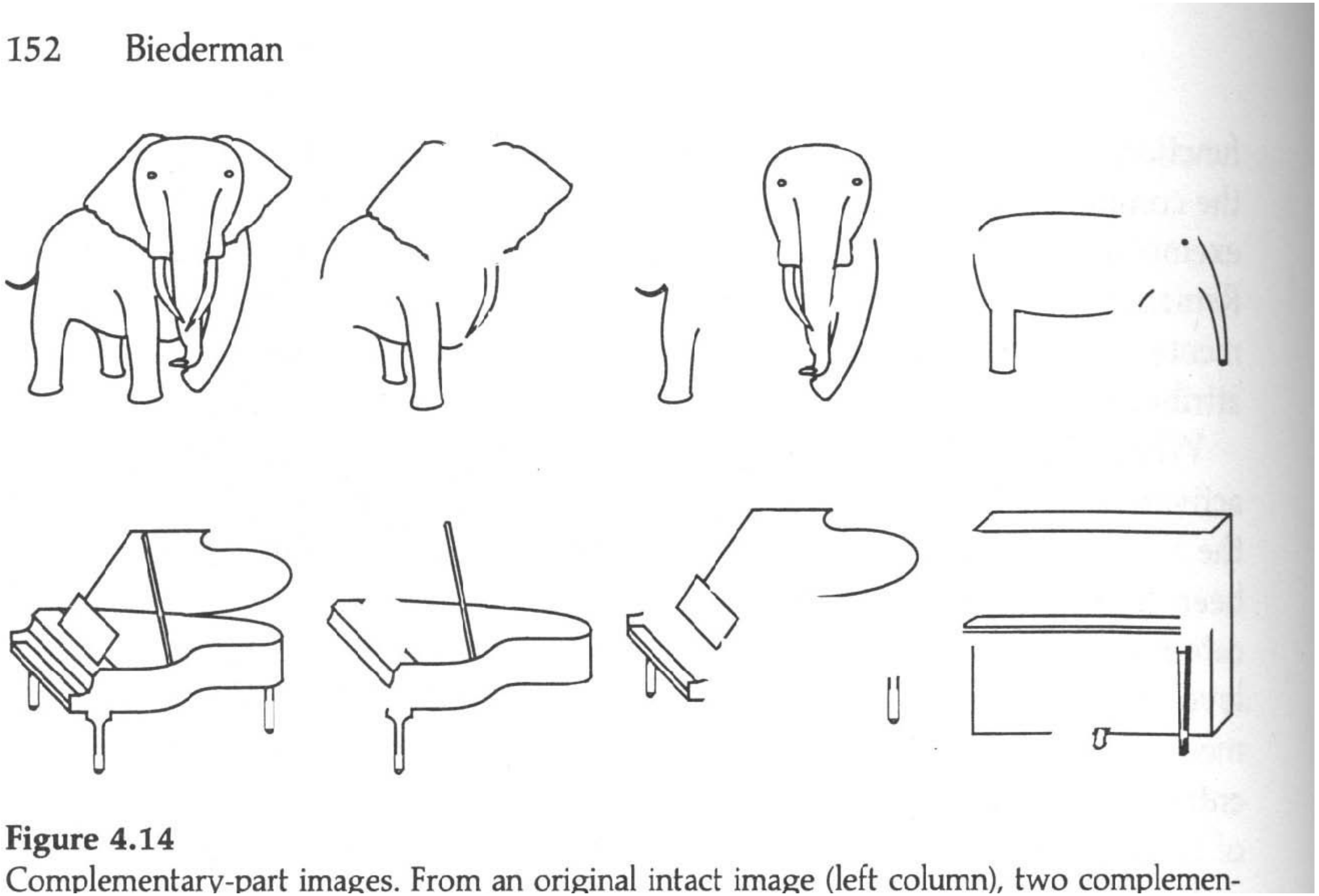
We know edges are special from human (mammalian) vision studies

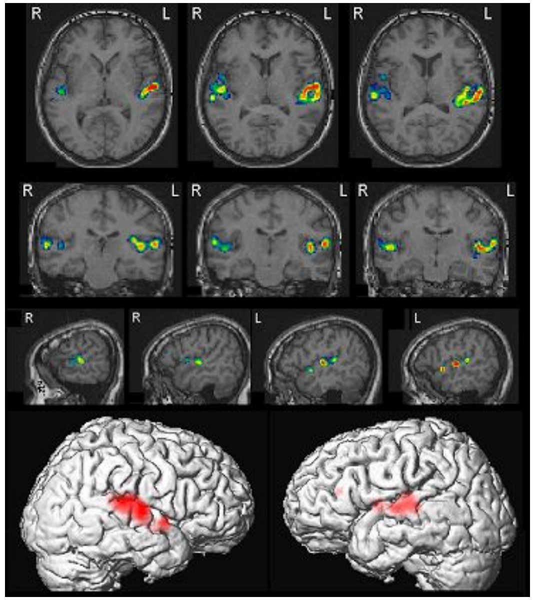


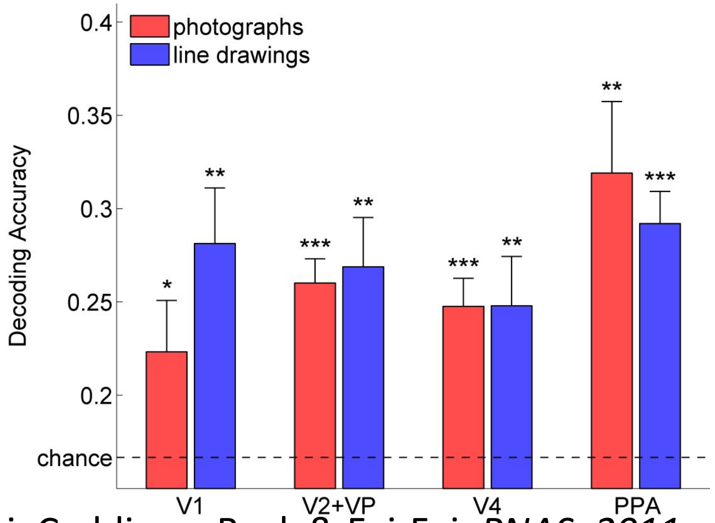


Hubel & Wiesel, 1960s

We know edges are special from human (mammalian) vision studies





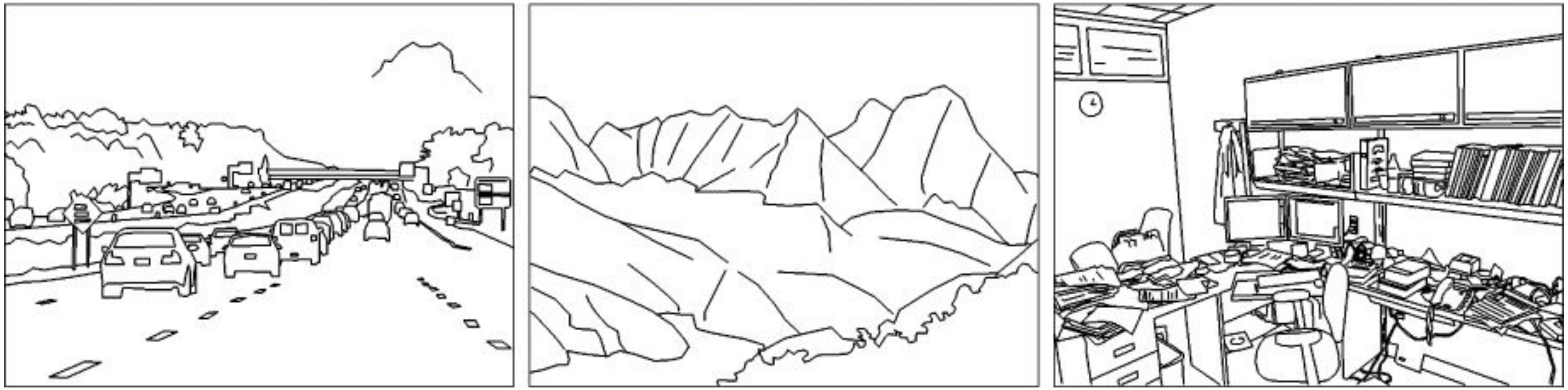


Walther, Chai, Caddigan, Beck & Fei-Fei, *PNAS, 2011*

Edge detection

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels

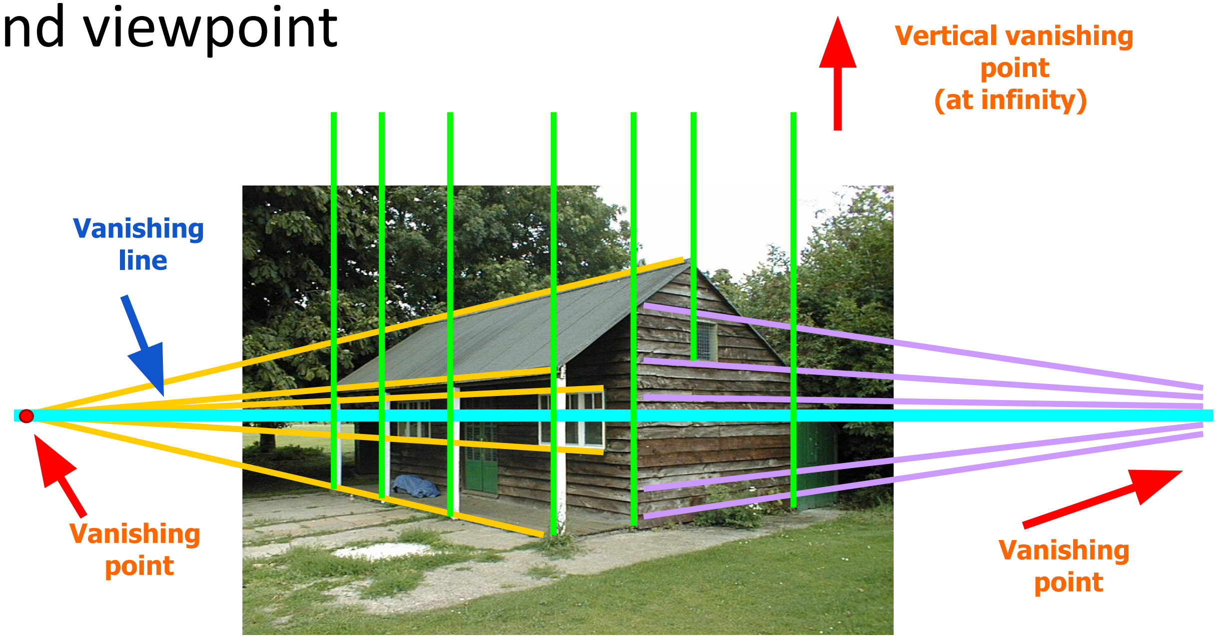
- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)



Why do we care about edges?

- Extract information, recognize objects

- Recover geometry and viewpoint



Source: J. Hayes

Origins of edges

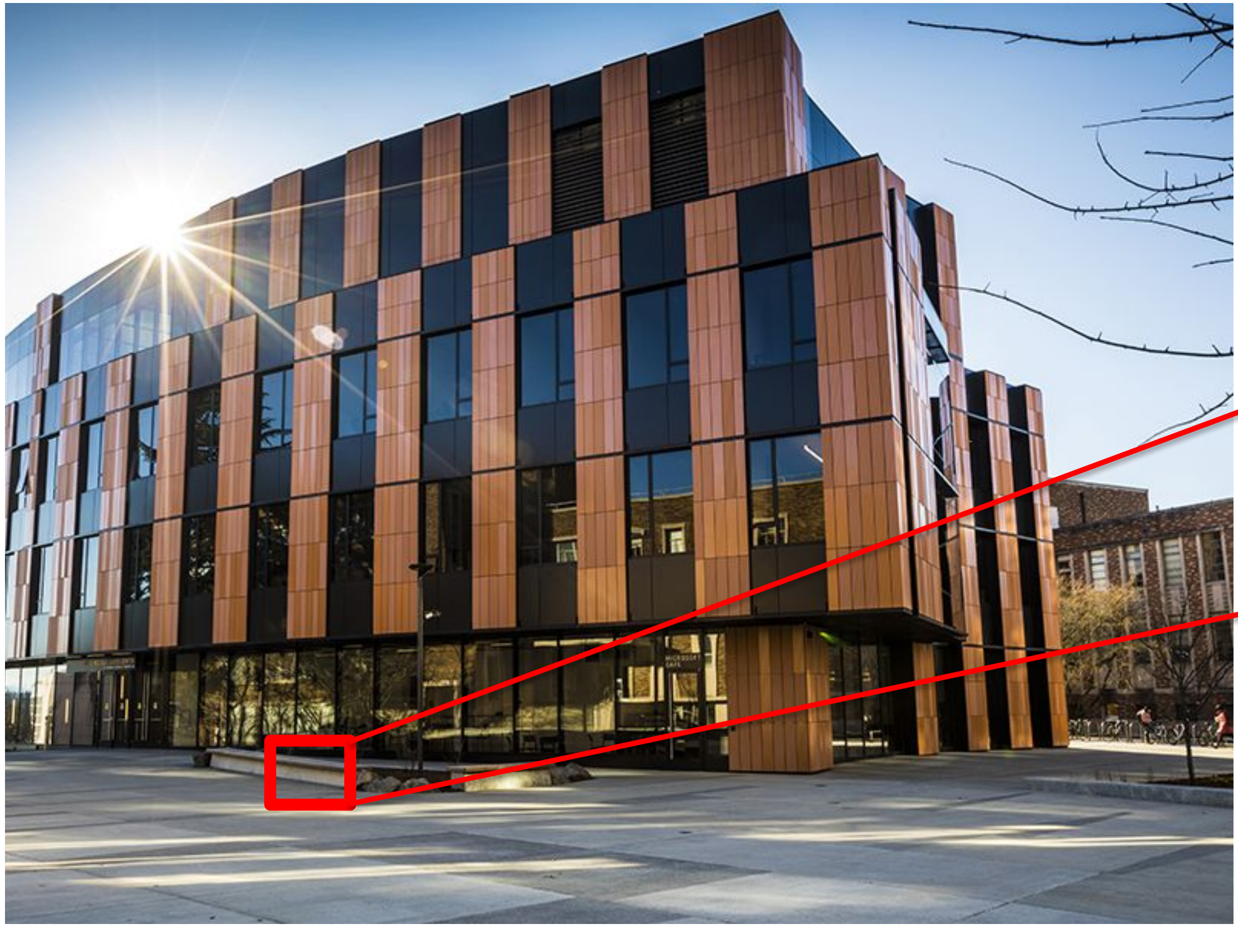
surface normal discontinuity

depth discontinuity

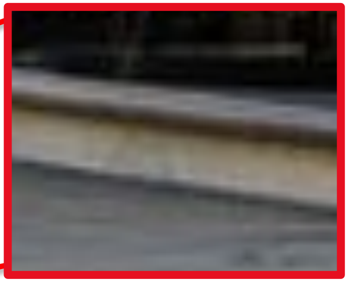
surface color discontinuity

illumination discontinuity

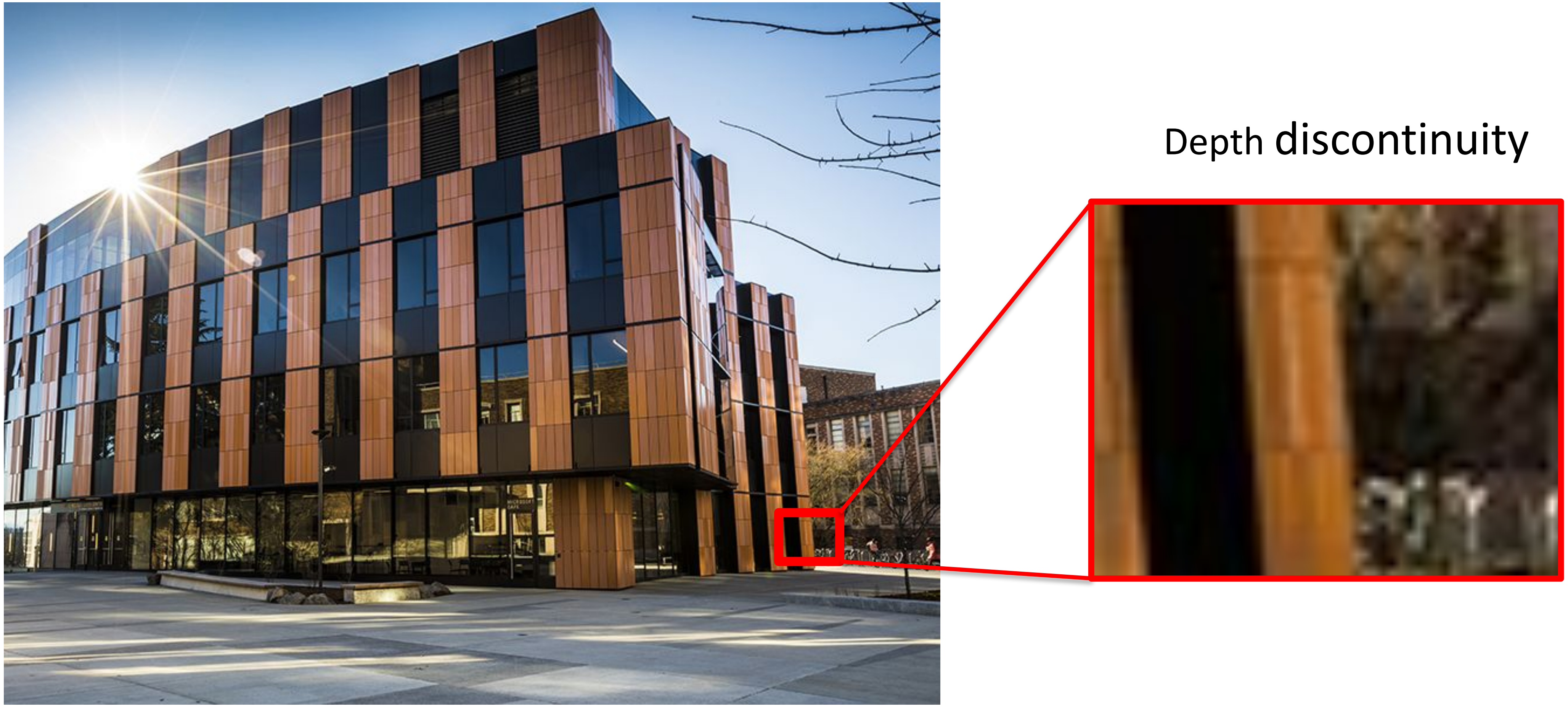
Closeup of edges



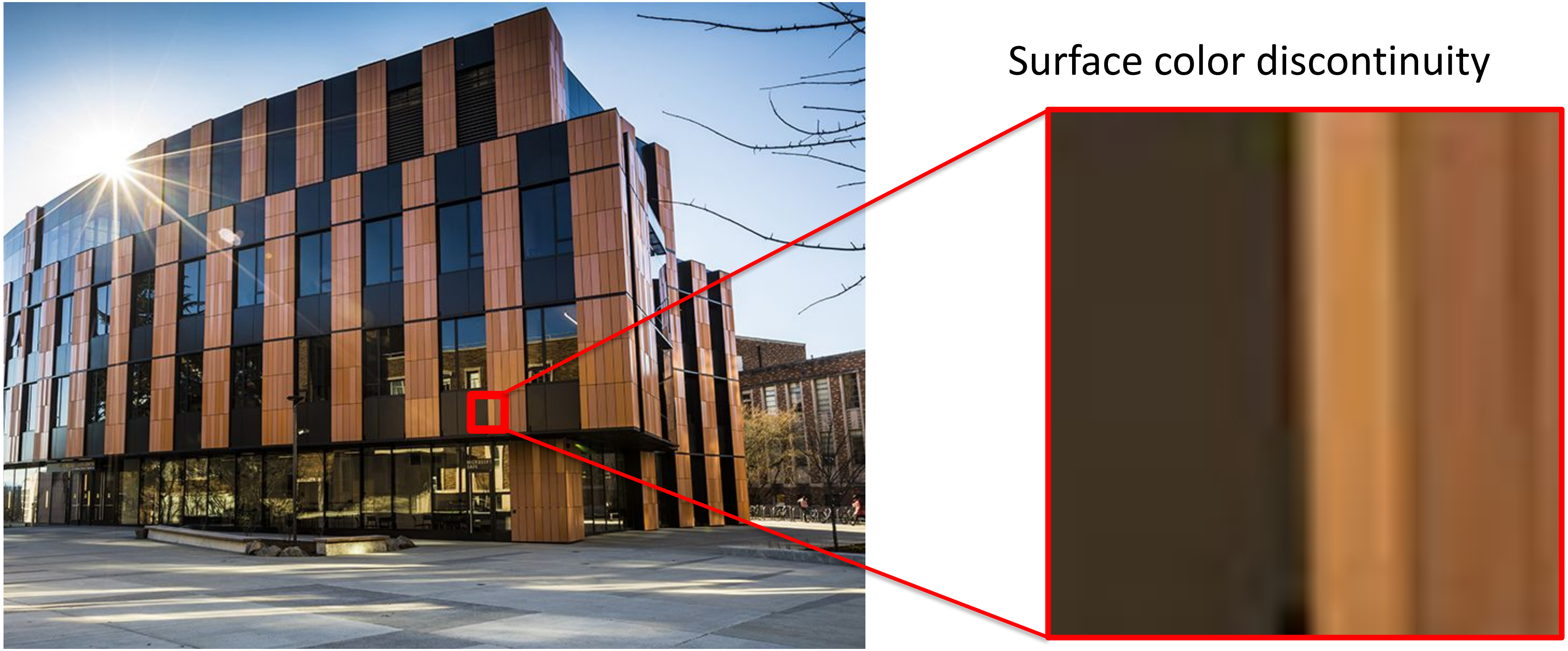
Surface normal discontinuity



Closeup of edges



Closeup of edges



Surface color discontinuity

What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector

Review: Derivatives in 1D - example

$$y = x^2 + x^4$$

Q. What is the dy/dx?

Review: Derivatives in 1D - example

$$y = x^2 + x^4$$

$$\frac{dy}{dx} = 2x + 4x^3$$

Derivatives in 1D - example

$$y = x^2 + x^4$$

$$\frac{dy}{dx} = 2x + 4x^3$$

$$y = \sin x + e^{-x}$$

Q. What is the dy/dx?

Derivatives in 1D - example

$$y = x^2 + x^4$$

$$\frac{dy}{dx} = 2x + 4x^3$$

$$y = \sin x + e^{-x}$$

$$\frac{dy}{dx} = \cos x + (-1)e^{-x}$$

Approximating derivatives using numerical differentiation

$$\frac{df}{dx} = \lim_{\triangle x = 0} \frac{f[x + \triangle x] - f[x]}{\triangle x} = f'(x) = f_x$$

Approximating derivatives using numerical differentiation

Change in *f* at *x*

$$\frac{df}{dx} = \lim_{\triangle x = 0} \frac{f[x + \triangle x] - f[x]}{\triangle x} = f'(x) = f_x$$

Change in *x*

In discrete derivatives with images, smallest value of x is 1 pixel

$$\frac{df}{dx} = \lim_{\triangle x = 0} \frac{f[x + \triangle x] - f[x]}{\triangle x} = f'(x) = f_x$$

$$= \frac{f[x+1] - f[x]}{1}$$

$$= f[x+1] - f[x]$$

This is called a forward derivative

But change at x can be measured in many different ways

$$\frac{df}{dx} = f[x] - f[x-1]$$

Backward

But change at x can be measured in many different ways

$$\frac{df}{dx} = f[x] - f[x-1]$$ Backward

$$= f[x+1] - f[x]$$ Forward

But change at x can be measured in many different ways

$$\frac{df}{dx} = f[x] - f[x-1]$$ Backward

$$= f[x+1] - f[x]$$ Forward

$$= \frac{1}{2}(f[x+1] - f[x-1])$$ Central

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = ??$$

Q. What is the equation in width (2nd) dimension?

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n, m] = f[n, m] - f[n, m-1]$$

Q. Let's write this as a filter

Remember the moving average filter:

$$h[\cdot,\cdot] = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

?	?	?
?	**?**	?
?	?	?

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

?	?	?
?	**1**	?
?	?	?

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

?	?	?
?	1	?
?	?	?

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

0	?	?
?	1	?
?	?	?

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n, m] = f[n, m] - f[n, m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

0	?	?
?	1	?
?	?	?

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

0	**0**	**0**
?	1	?
?	?	?

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

0	0	0
?	1	?
?	?	?

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Let's write this as a filter

$h[\cdot,\cdot]$

0	0	0
?	1	?
0	**0**	**0**

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Last ones: What are these two?

$h[\cdot,\cdot]$

0	0	0
?	1	?
0	0	0

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation

$$g[n,m] = f[n,m] - f[n,m-1]$$

Q. Last ones: What are these two?

$h[\cdot,\cdot]$

0	0	0
0	1	**-1**
0	0	0

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

Remember the moving average filter:

$h[\cdot,\cdot]$

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Designing filters that perform differentiation

- Using Backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

- Using Forward differentiation:

Q. What is the formula?

Designing filters that perform differentiation

- Using Backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

- Using Forward differentiation:

?	?	?

$$g[n,m] = f[n,m+1] - f[n,m]$$

Q. What is the filter look like?

Designing filters that perform differentiation

- Using Backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

- Using Forward differentiation:

1	**-1**	**0**

$$g[n,m] = f[n,m+1] - f[n,m]$$

Designing filters that perform differentiation

- Using Backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

- Using Forward differentiation:

1	-1	0

$$g[n,m] = f[n,m+1] - f[n,m]$$

- Using Central differentiation:

Q. What is the formula?

Designing filters that perform differentiation

- Using Backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

- Using Forward differentiation:

1	-1	0

$$g[n,m] = f[n,m+1] - f[n,m]$$

- Using Central differentiation:

?	?	?

Q. What is the filter?

$$g[n,m] = f[n,m+1] - f[n,m-1]$$

Designing filters that perform differentiation

- Using Backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

- Using Forward differentiation:

1	-1	0

$$g[n,m] = f[n,m+1] - f[n,m]$$

- Using Central differentiation:

1	**0**	**-1**

$$g[n,m] = f[n,m+1] - f[n,m-1]$$

Derivative in width dimension for one row

Using backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

$$f[0,:] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

Derivative in width dimension for one row

Using backward differentiation:

0	1	-1

$$g[n, m] = f[n, m] - f[n, m-1]$$

$$f[0, :] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

$$\frac{df}{dm}[0, :] = [\ ? \ ]$$

Derivative in width dimension for one row

Using backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

$$f[0,:] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

$$\frac{df}{dm}[0,:] = [10, \quad ? \quad]$$

Derivative in width dimension for one row

Using backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

$$f[0,:] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

$$\frac{df}{dm}[0,:] = [10, \quad 5, \quad ? \quad]$$

Derivative in width dimension for one row

Using backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

$$f[0,:] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

$$\frac{df}{dm}[0,:] = [10, \quad 5, \quad -5, \quad ? \quad]$$

Derivative in width dimension for one row

Using backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

$$f[0,:] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

$$\frac{df}{dm}[0,:] = [10, \quad 5, \quad -5, \quad 0, \quad ? \qquad]$$

Derivative in width dimension for one row

0	1	-1

Using backward differentiation:

$$g[n,m] = f[n,m] - f[n,m-1]$$

$$f[0,:] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

$$\frac{df}{dm}[0,:] = [10, \quad 5, \; -5, \quad 0, \quad 15, \quad ? \quad ? \quad ?\;]$$

Derivative in width dimension for one row

Using backward differentiation:

0	1	-1

$$g[n,m] = f[n,m] - f[n,m-1]$$

$$f[0,:] = [10, \quad 15, \quad 10, \quad 10, \quad 25, \quad 20, \quad 20, \quad 20]$$

$$\frac{df}{dm}[0,:] = [10, \quad 5, \quad -5, \quad 0, \quad 15, \quad -5, \quad 0, \quad 0]$$

Discrete derivation in 2D:

$$\text{Given function } f[n, m]$$

$$\text{Gradient filter } \nabla f[n, m] = \begin{bmatrix} \frac{df}{dn} \\ \frac{df}{dm} \end{bmatrix} = \begin{bmatrix} f_n \\ f_m \end{bmatrix}$$

Discrete derivation in 2D:

$$\text{Given function } f[n,m]$$

$$\text{Gradient filter } \nabla f[n,m] = \begin{bmatrix} \frac{df}{dn} \\ \frac{df}{dm} \end{bmatrix} = \begin{bmatrix} f_n \\ f_m \end{bmatrix}$$

$$\text{Gradient magnitude } |\nabla f[n,m]| = \sqrt{f_n^2 + f_m^2}$$

Discrete derivation in 2D:

$$\text{Given function } f[n,m]$$

$$\text{Gradient filter } \nabla f[n,m] = \begin{bmatrix} \frac{df}{dn} \\ \frac{df}{dm} \end{bmatrix} = \begin{bmatrix} f_n \\ f_m \end{bmatrix}$$

$$\text{Gradient magnitude } |\nabla f[n,m]| = \sqrt{f_n^2 + f_m^2}$$

$$\text{Gradient direction } \theta = \tan^{-1}(\frac{f_m}{f_n})$$

2D discrete derivative filters

Q. What does this filter do?

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

2D discrete derivative filters

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

Q. What does this filter do?

$$h[n,m] = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

2D discrete derivative - example

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

2D discrete derivative - example

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} ? & ? & ? & ? & ? \\ & & & & \\ & & & & \end{bmatrix}$$

2D discrete derivative - example

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \qquad h[n,m] = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ ? & ? & ? & ? & ? \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

2D discrete derivative - example

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \qquad h[n,m] = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 0 & 0 & 0 & 0 & 0 \\ ? & ? & ? & ? & ? \\ & & & & \\ & & & & \end{bmatrix}$$

2D discrete derivative - example

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ ? & ? & ? & ? & ? \\ & & & & \end{bmatrix}$$

2D discrete derivative - example

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ ? & ? & ? & ? & ? \end{bmatrix}$$

2D discrete derivative - example

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -10 & -10 & -20 & -20 & -20 \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} ? & & & & \\ ? & & & & \\ ? & & & & \\ ? & & & & \\ ? & & & & \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & ? & & & \\ 10 & ? & & & \\ 10 & ? & & & \\ 10 & ? & & & \\ 10 & ? & & & \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & ? & & \\ 10 & 10 & ? & & \\ 10 & 10 & ? & & \\ 10 & 10 & ? & & \\ 10 & 10 & ? & & \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 10 & ? & \\ 10 & 10 & 10 & ? & \\ 10 & 10 & 10 & ? & \\ 10 & 10 & 10 & ? & \\ 10 & 10 & 10 & ? & \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \\ 10 & 10 & 10 & 0 & ? \end{bmatrix}$$

Let's do the other one

$$f[n,m] = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$$

$$h[n,m] = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

$$g[n,m] = \begin{bmatrix} 10 & 10 & 10 & 0 & -20 \\ 10 & 10 & 10 & 0 & -20 \\ 10 & 10 & 10 & 0 & -20 \\ 10 & 10 & 10 & 0 & -20 \\ 10 & 10 & 10 & 0 & -20 \end{bmatrix}$$

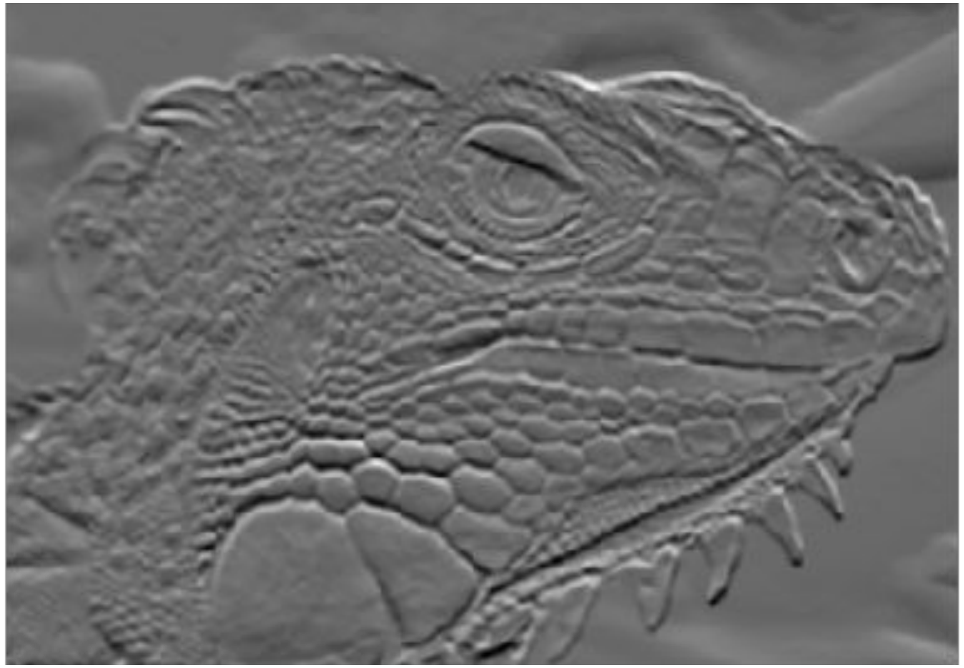
Q. Which filter was applied?

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

A

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

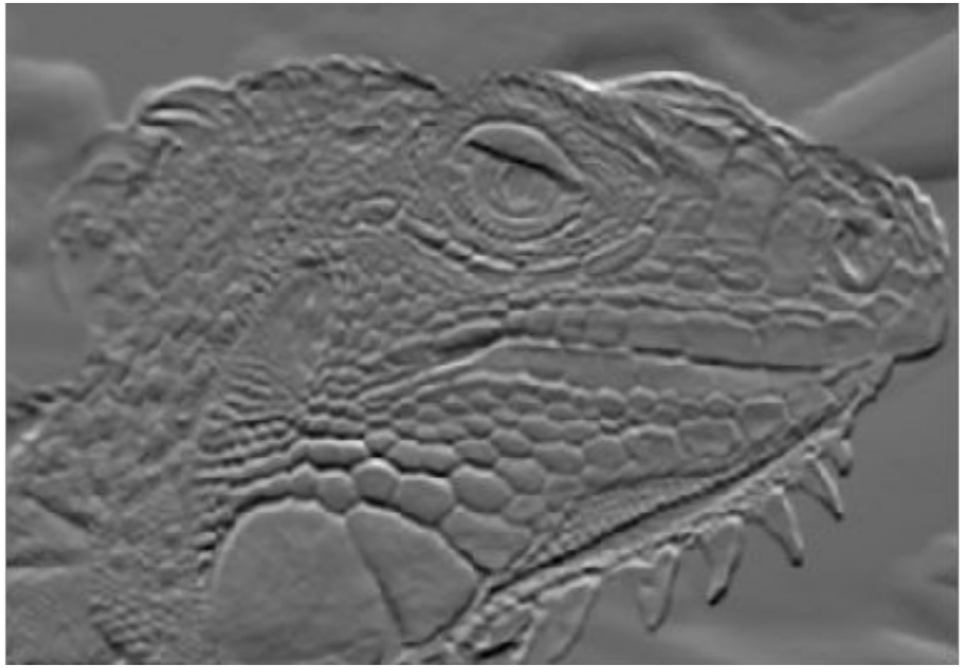
B

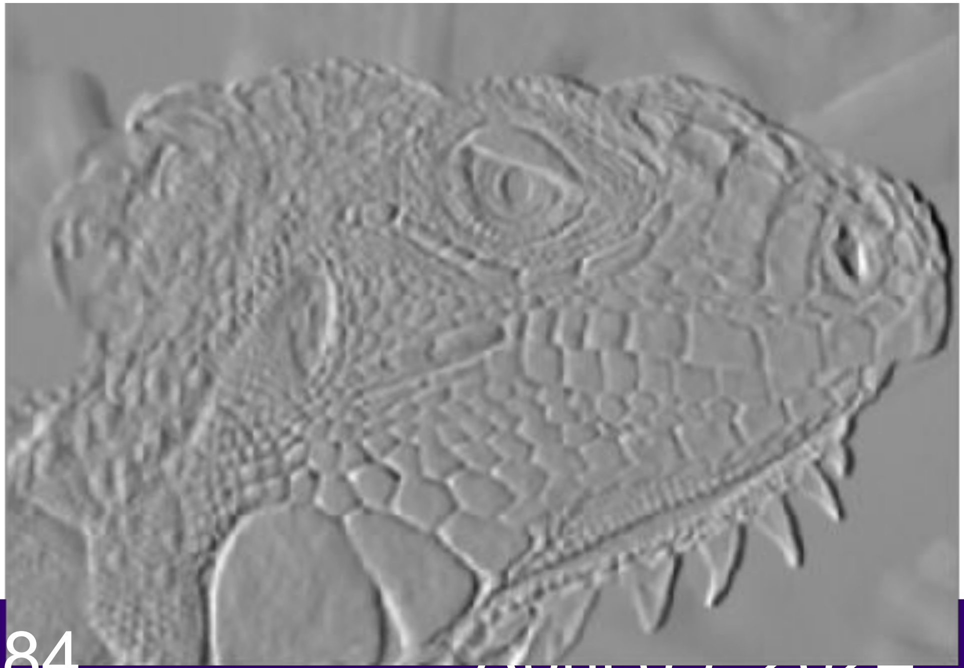


Q. Which filter was applied?

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

A **B**





What we will learn today

- Edge detection
- Image Gradients
- A simple edge detector

Characterizing edges

An edge is a place of rapid change in the image intensity function

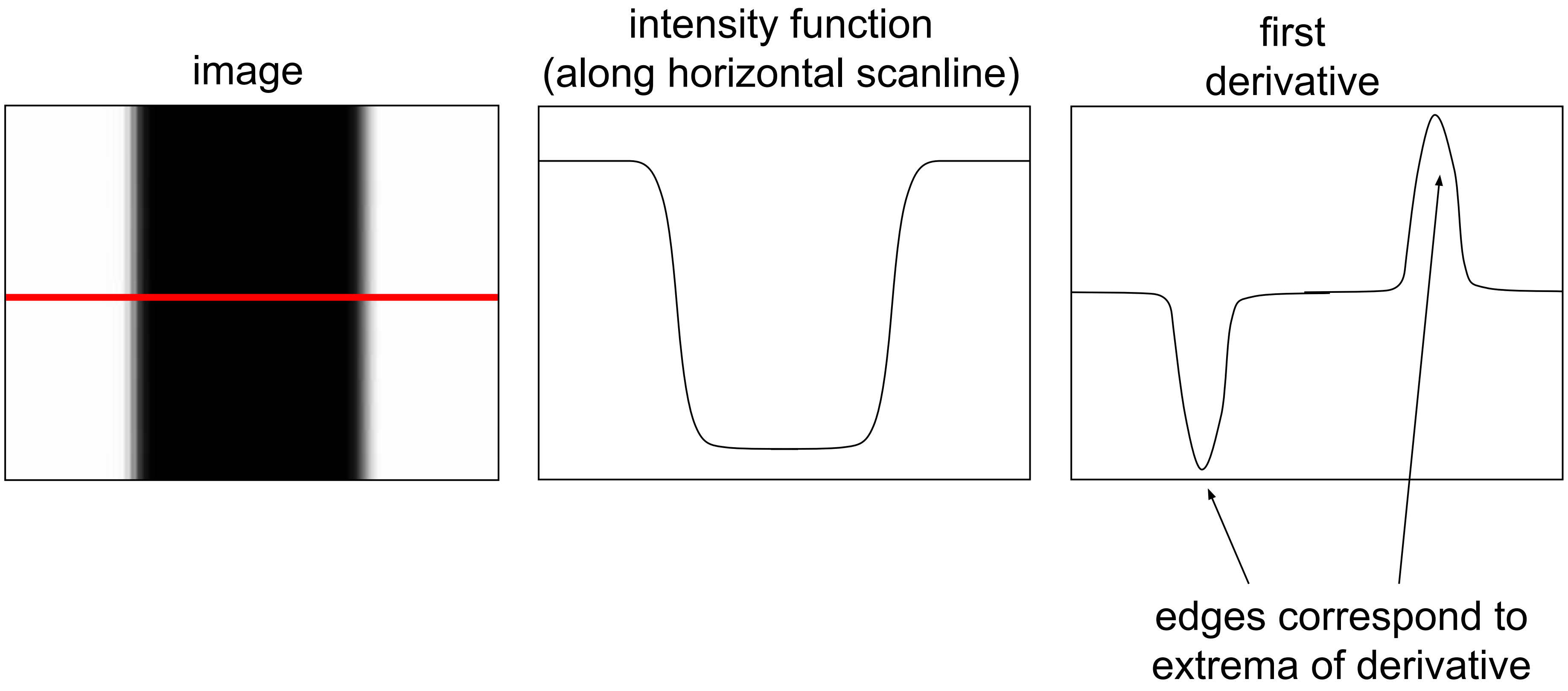
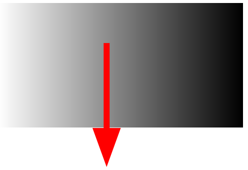
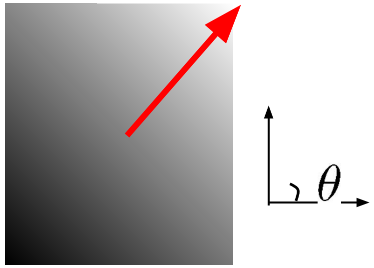


Image gradient

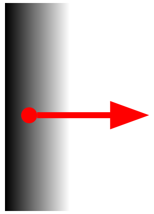
The gradient of an image:



$$\nabla_n f[n,m] = \begin{bmatrix} \frac{df}{dn} & 0 \end{bmatrix}$$



$$\nabla f[n,m] = \begin{bmatrix} \frac{df}{dn} & \frac{df}{dm} \end{bmatrix}$$



$$\nabla_m f[n,m] = \begin{bmatrix} 0 & \frac{df}{dm} \end{bmatrix}$$

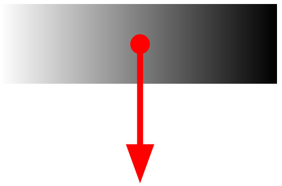
The gradient vector points in the direction of most rapid increase in intensity

$$\theta = \tan^{-1}\left(\frac{f_m}{f_n}\right)$$

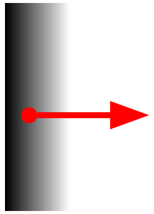
Source: Steve Seitz

Image gradient

The gradient of an image:

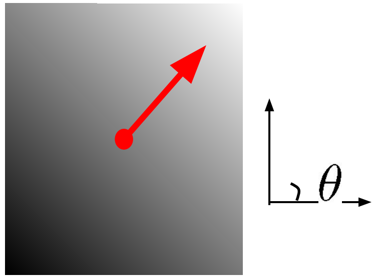


$$\nabla_n f[n,m] = \begin{bmatrix} \frac{df}{dn} & 0 \end{bmatrix}$$



$$\nabla_m f[n,m] = \begin{bmatrix} 0 & \frac{df}{dm} \end{bmatrix}$$

$$\nabla f[n,m] = \begin{bmatrix} \frac{df}{dn} & \frac{df}{dm} \end{bmatrix}$$



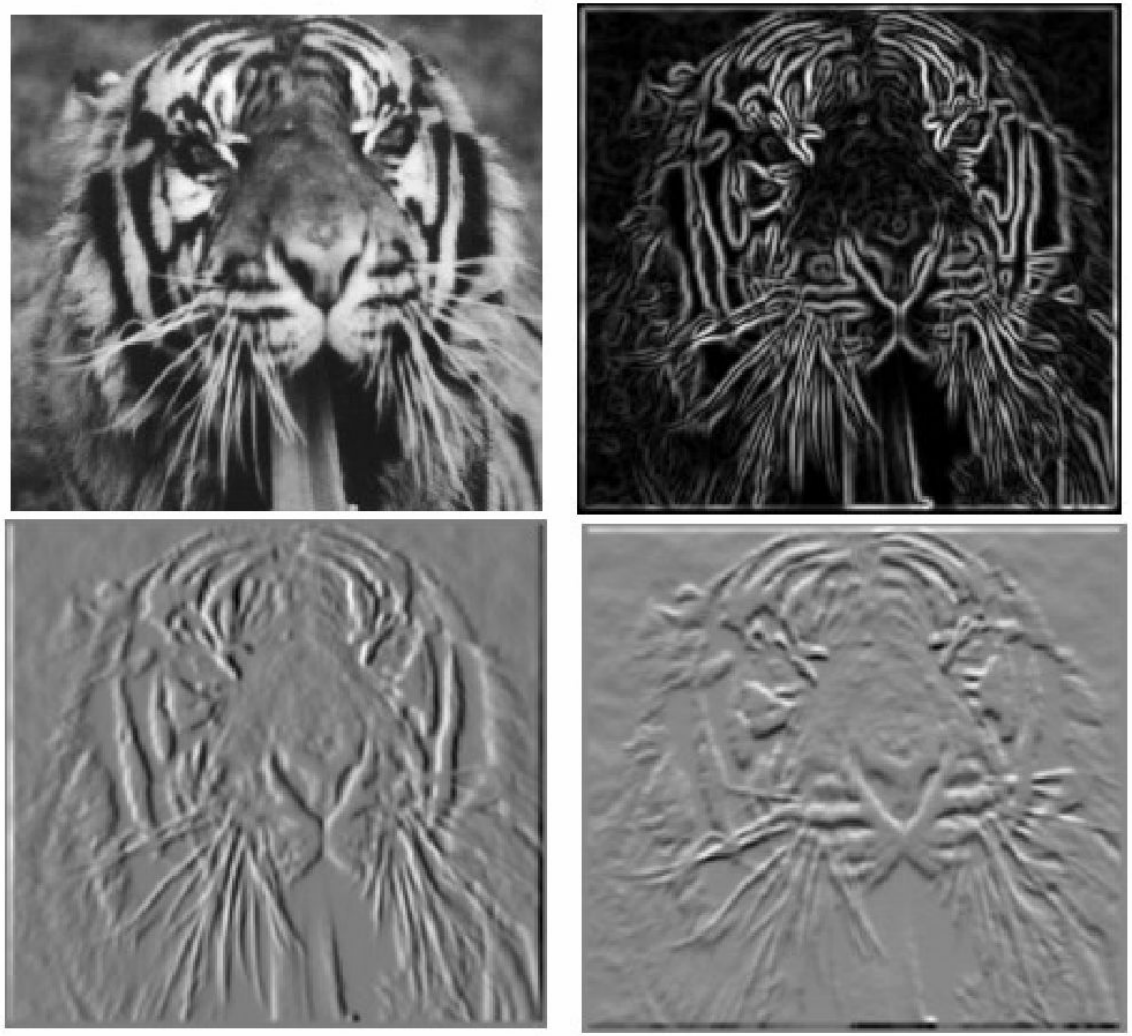
The gradient vector points in the direction of most rapid increase in intensity

$$\theta = \tan^{-1}\left(\frac{f_m}{f_n}\right)$$

The *edge strength* is given by the gradient magnitude

$$|\nabla f[n,m]| = \sqrt{f_n^2 + f_m^2}$$

Finite differences: example



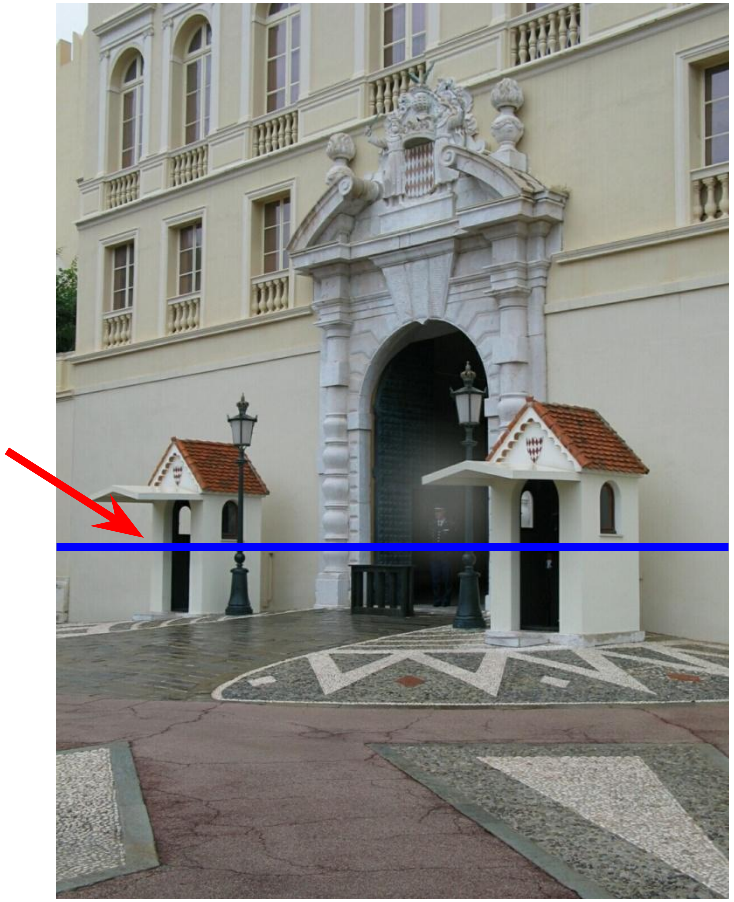
Original Image

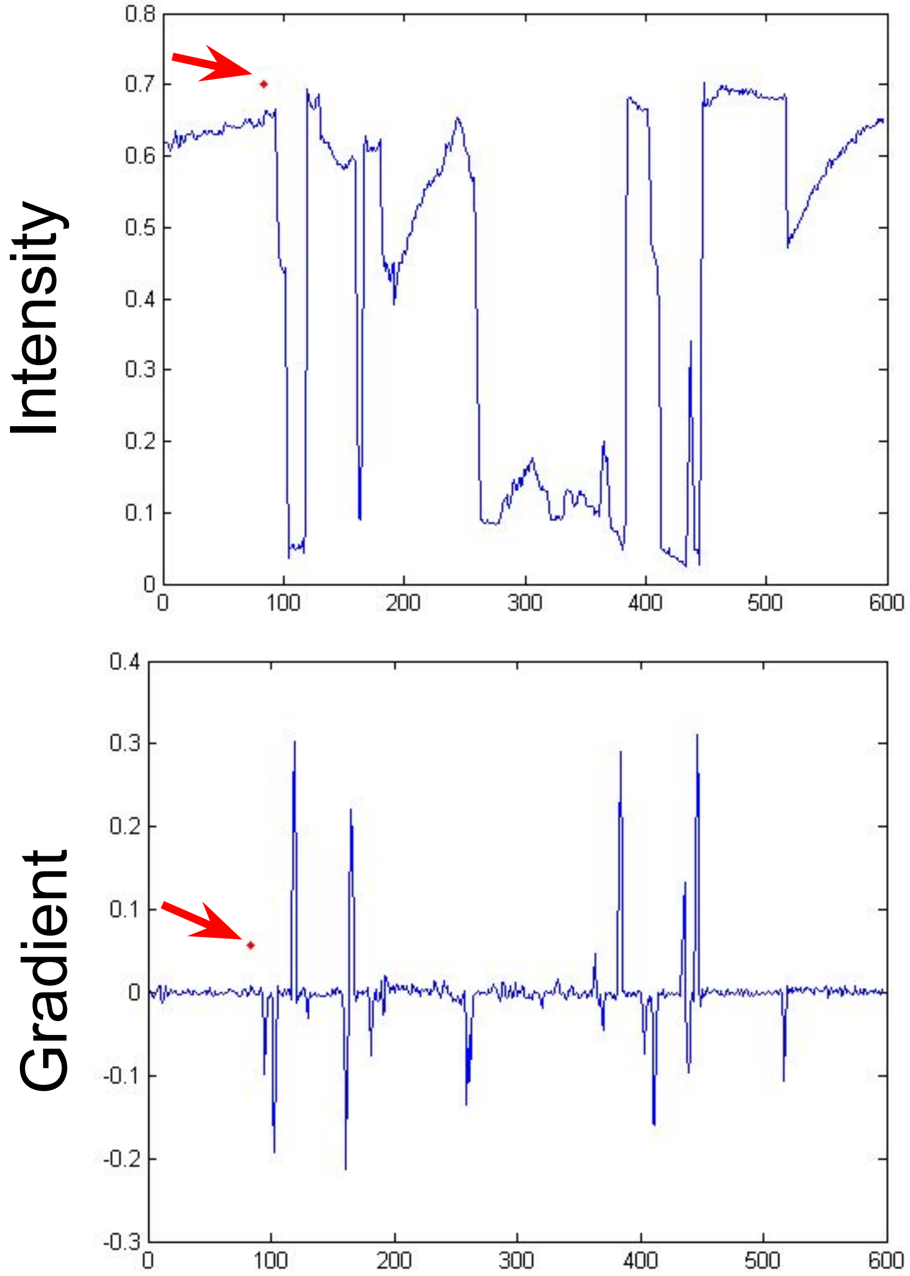
Gradient magnitude

width-direction

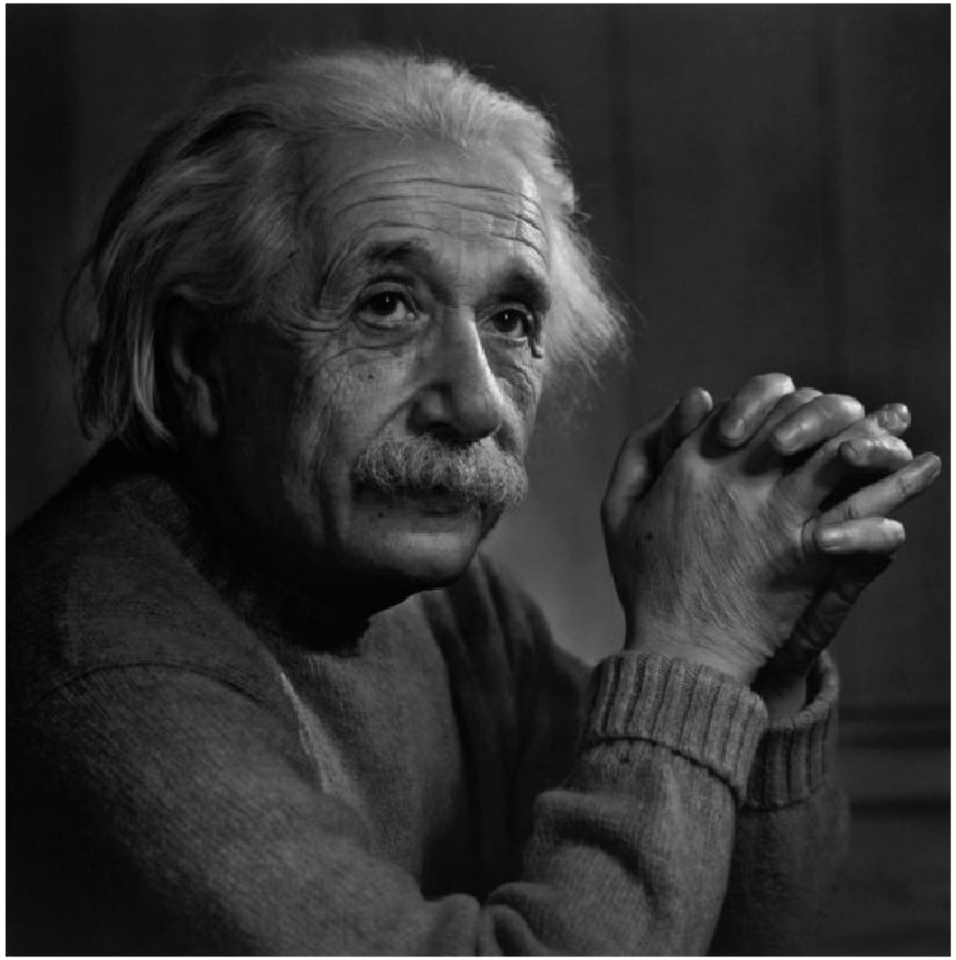
height-direction

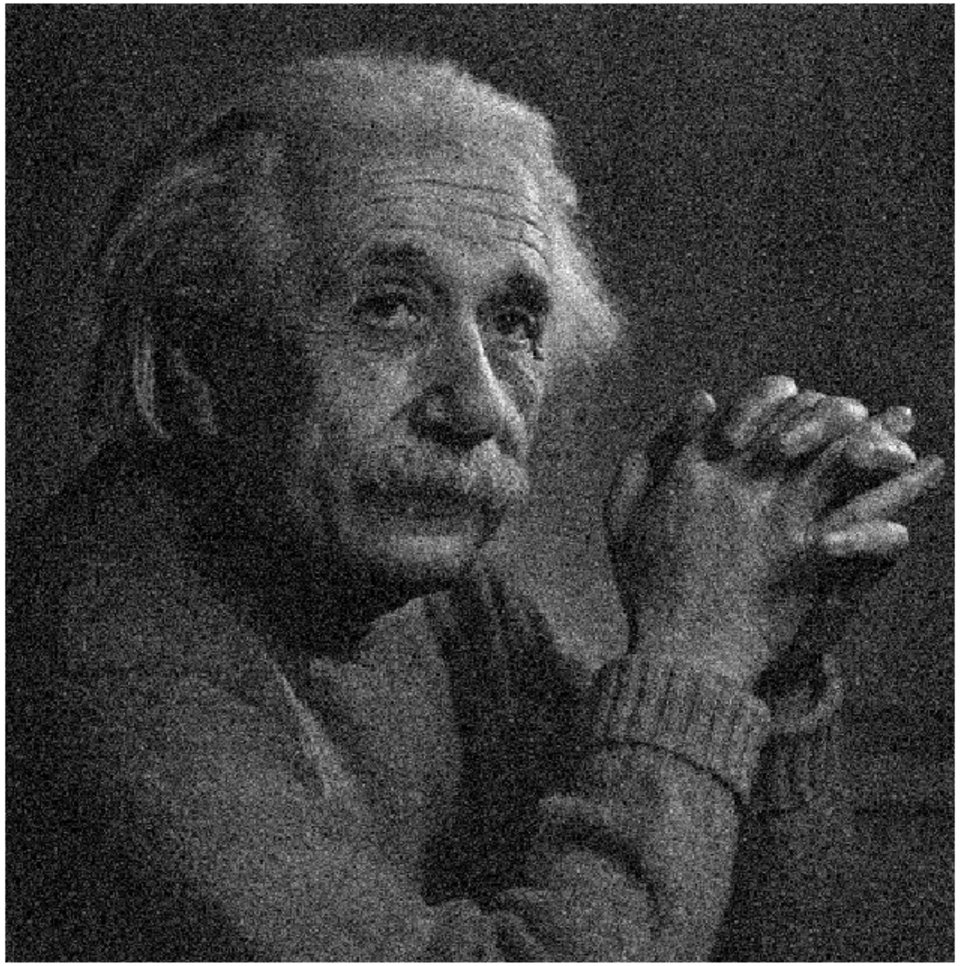
Intensity profile





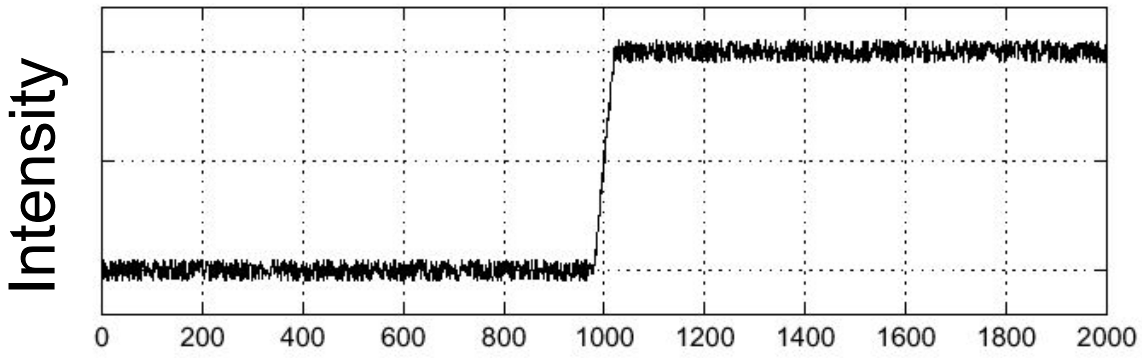
Q. What will happen if we use this edge detector on a noisy pixels?





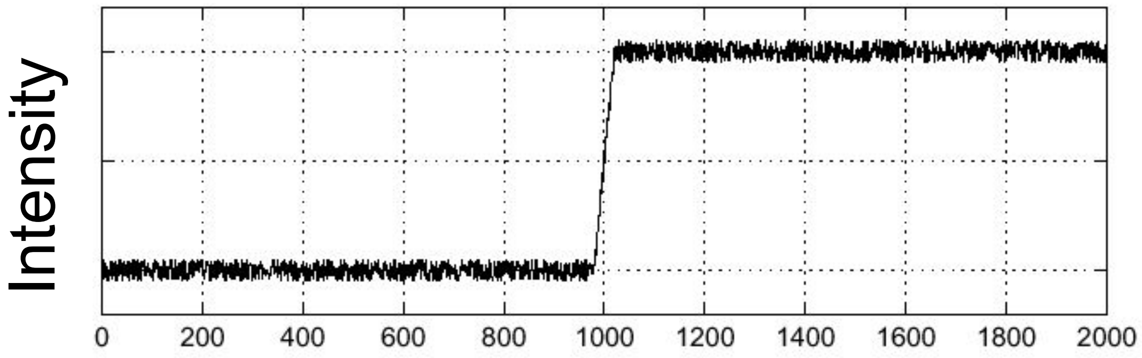
Effects of noise

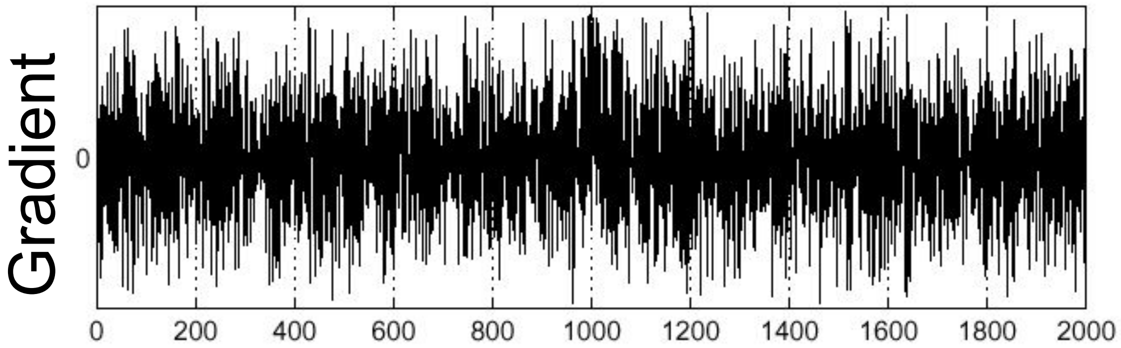
- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal



Effects of noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal





Where is the edge?

Source: S. Seitz

Effects of noise

- Finite difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- Q. What is a potential quick fix for noisy images?

Source: D. Forsyth

Effects of noise

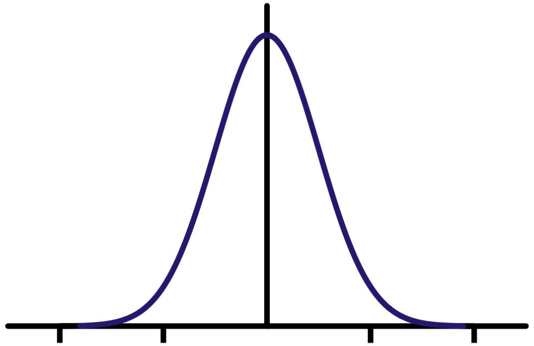
- Finite difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- Q. What is a potential quick fix for noisy images?
- Smoothing the image should help, by forcing pixels different to their neighbors (=noise pixels?) to look more like neighbors

Source: D. Forsyth

Smoothing with different filters

- Mean smoothing $\frac{1}{3}\begin{bmatrix}1\\1\\1\end{bmatrix}$ $\frac{1}{3}\begin{bmatrix}1 & 1 & 1\end{bmatrix}$

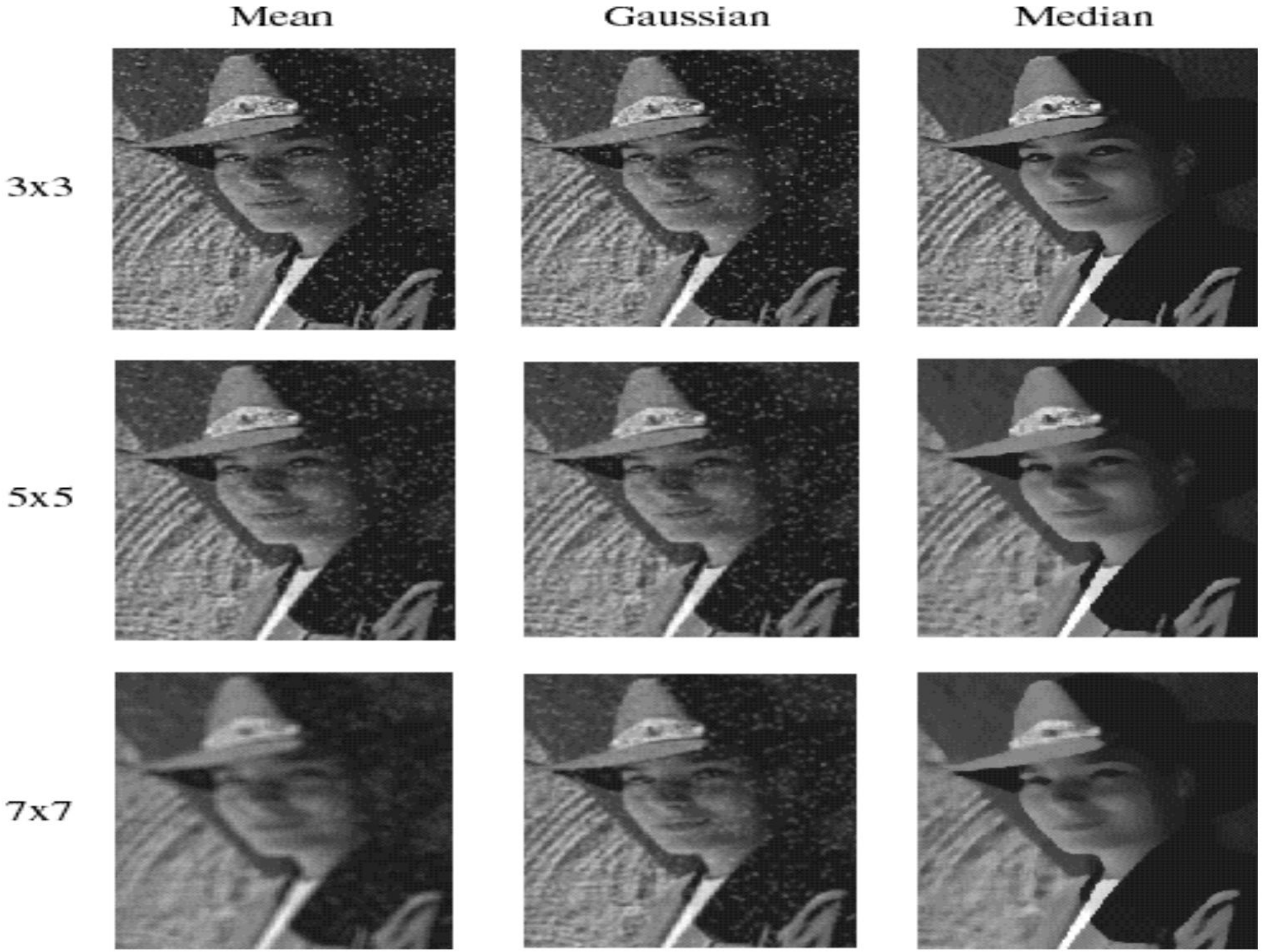
- Gaussian (smoothing * derivative)



$\frac{1}{4}\begin{bmatrix}1\\2\\1\end{bmatrix}$ $\frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix}$

Slide credit: Steve Seitz

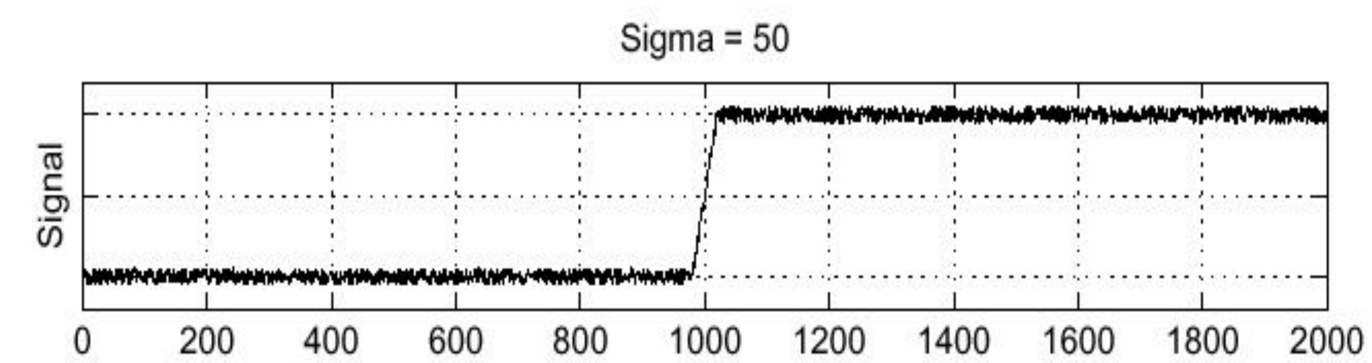
Smoothing with different filters



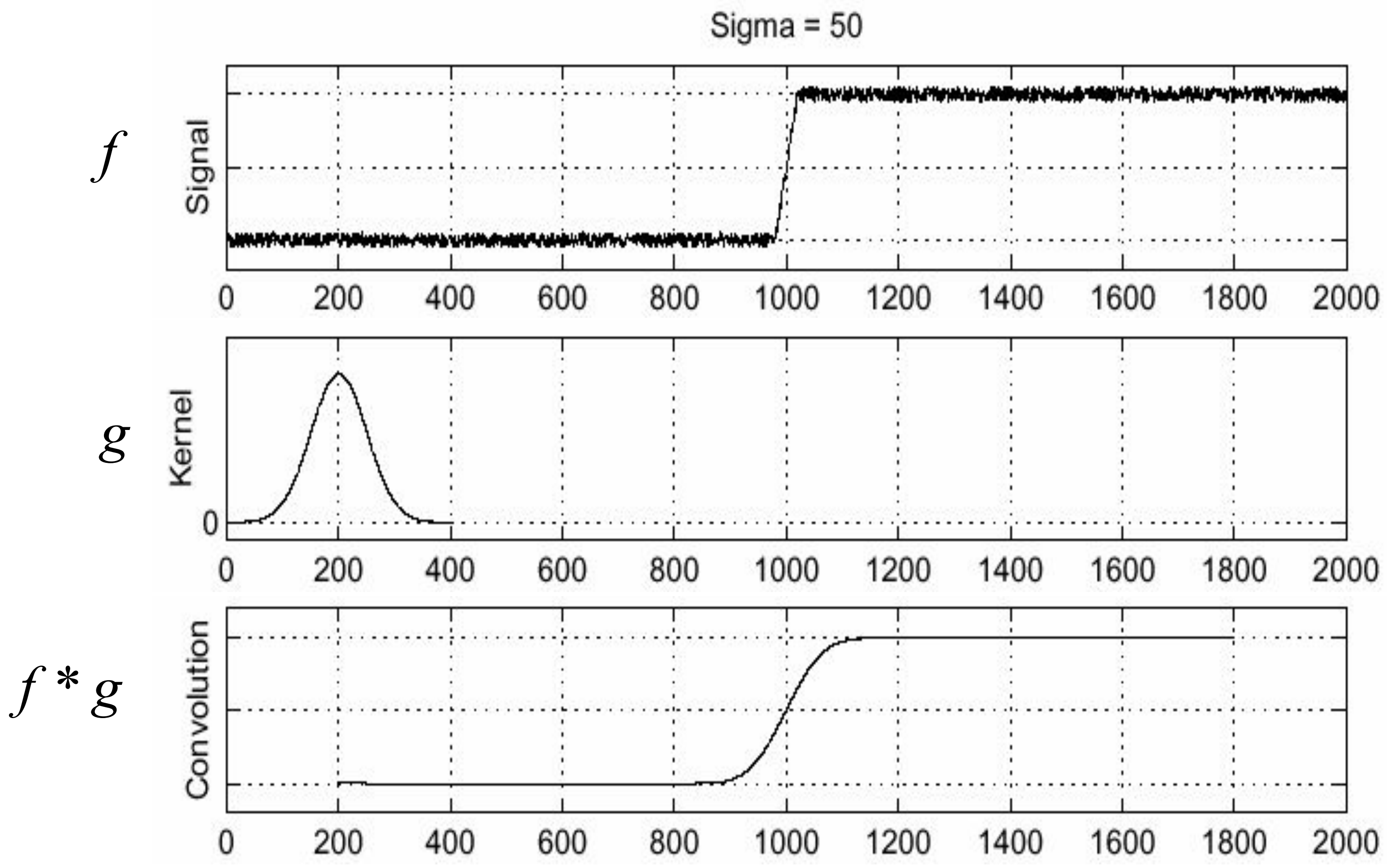
Slide credit: Steve Seitz

Solution: input function

f



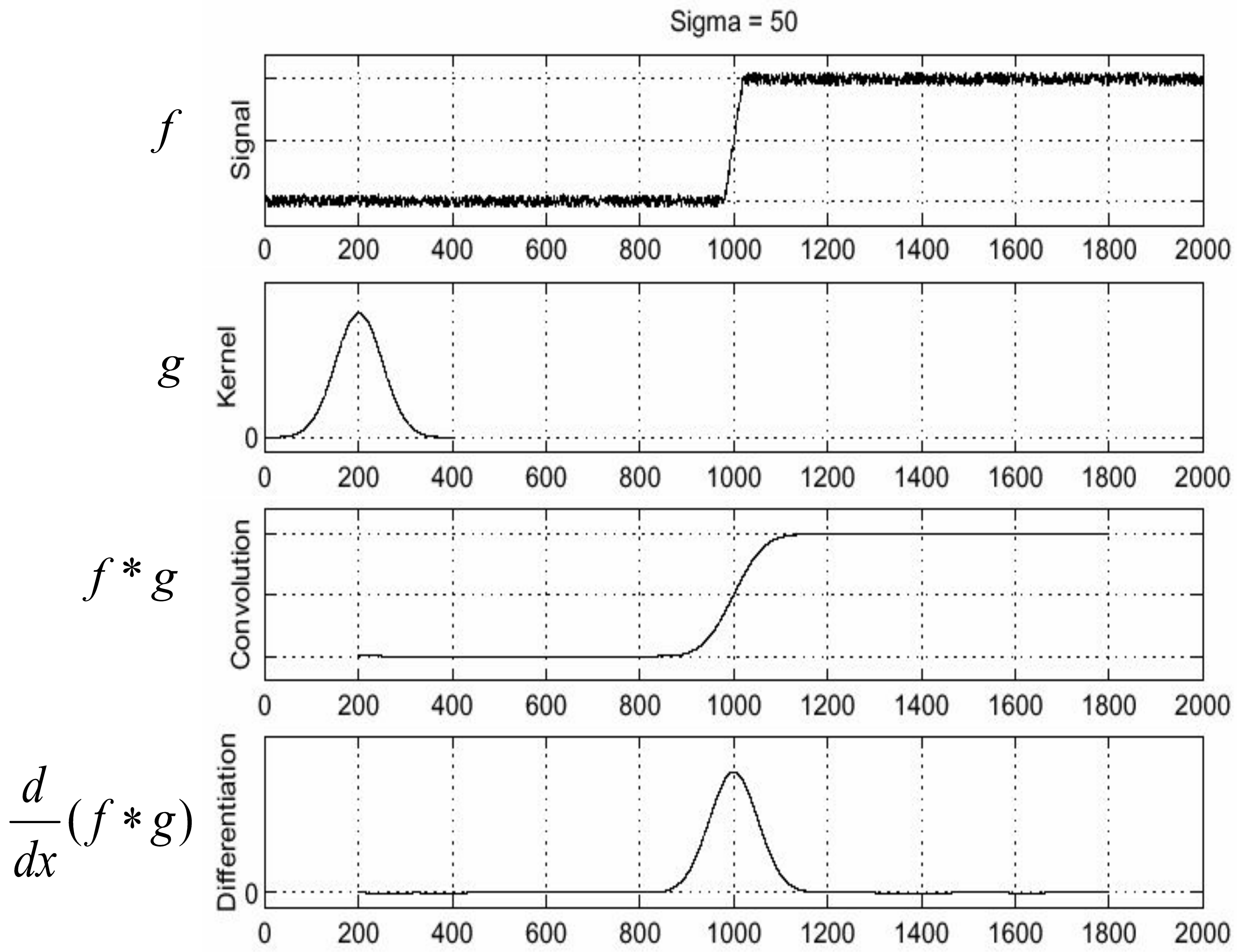
Solution: smooth first



Source: S. Seitz

Solution: smooth first

To find edges, look for peaks in $\frac{d}{dx}(f * g)$



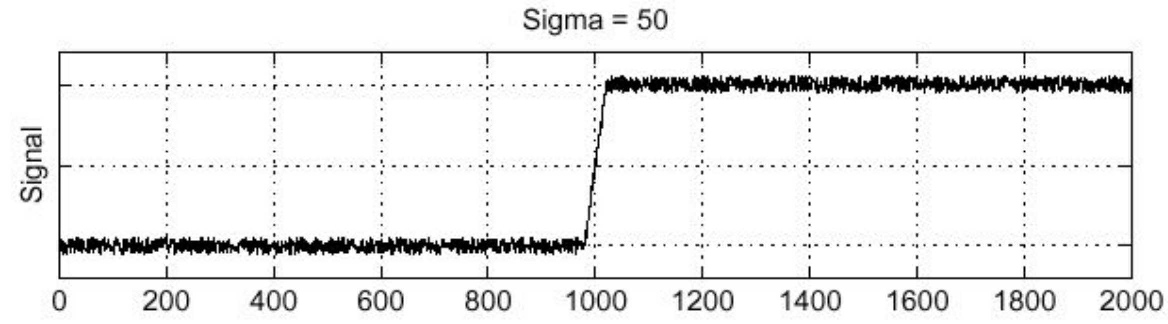
Source: S. Seitz

Derivative theorem of convolution

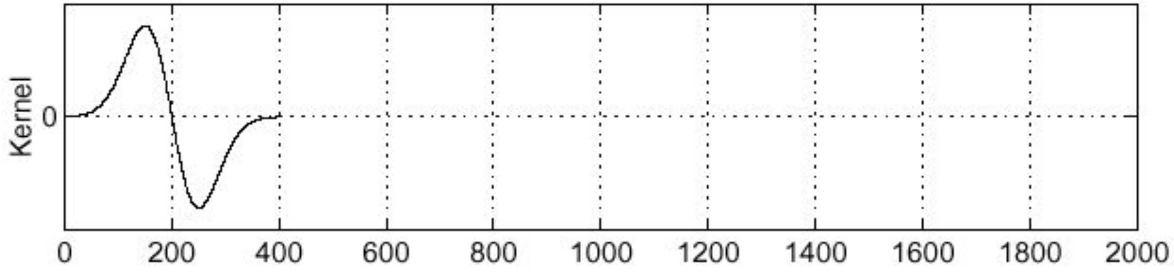
- This theorem gives us a very useful property:

$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

f



$\frac{d}{dx} g$



Source: S. Seitz

Derivative theorem of convolution

- This theorem gives us a very useful property:

$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

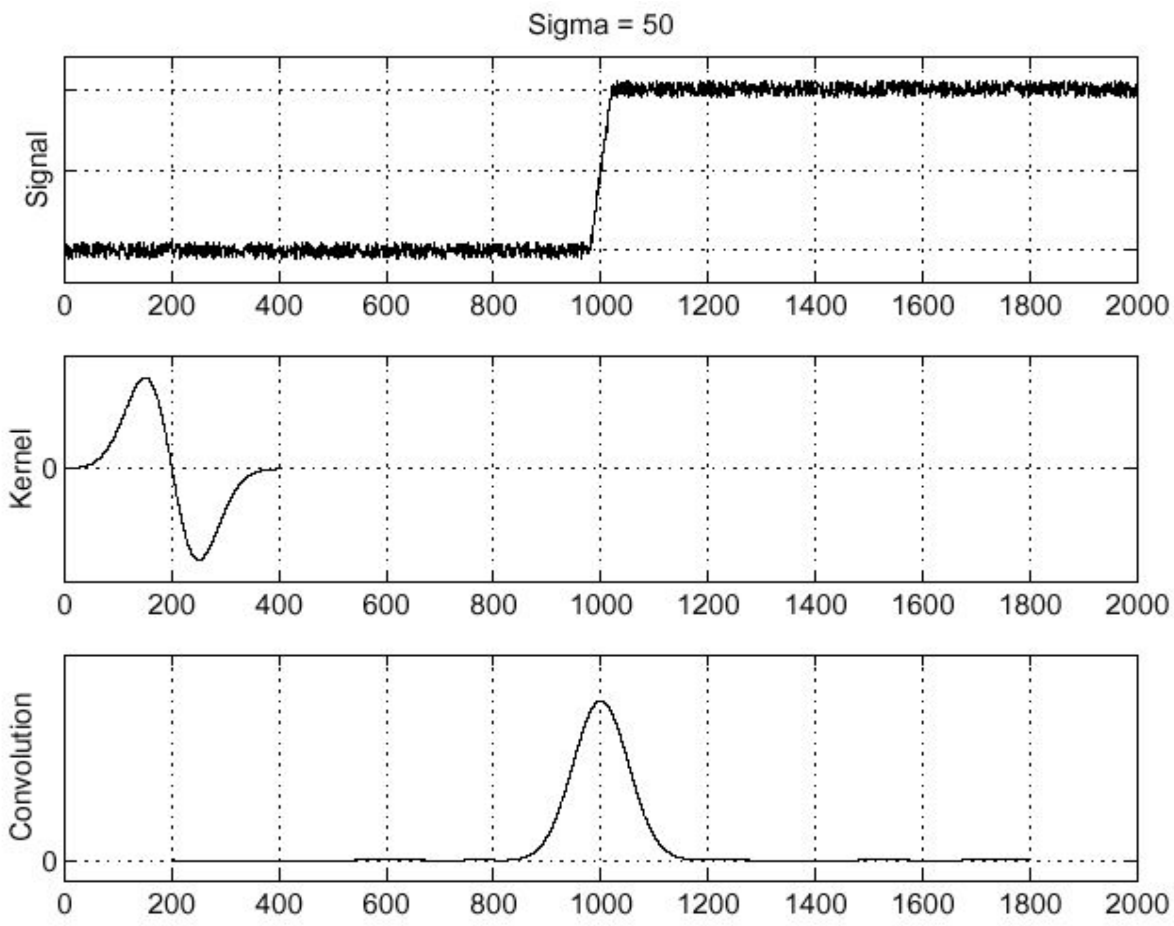
- This saves us one operation:

We can precompute: $\frac{d}{dx} g$

f

$\frac{d}{dx} g$

$f * \frac{d}{dx} g$



Source: S. Seitz

Derivative theorem of convolution

- This theorem gives us a very useful property:

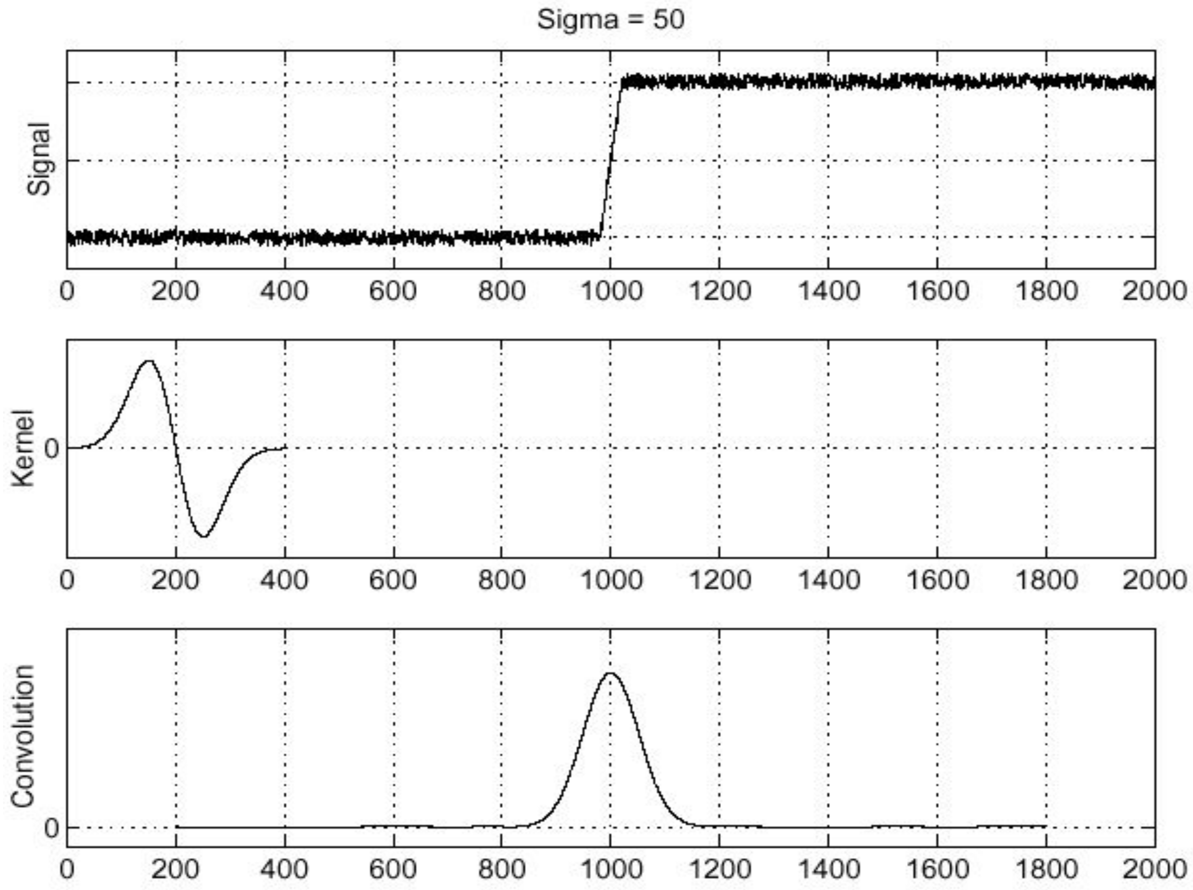
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

- This saves us one operation:

f

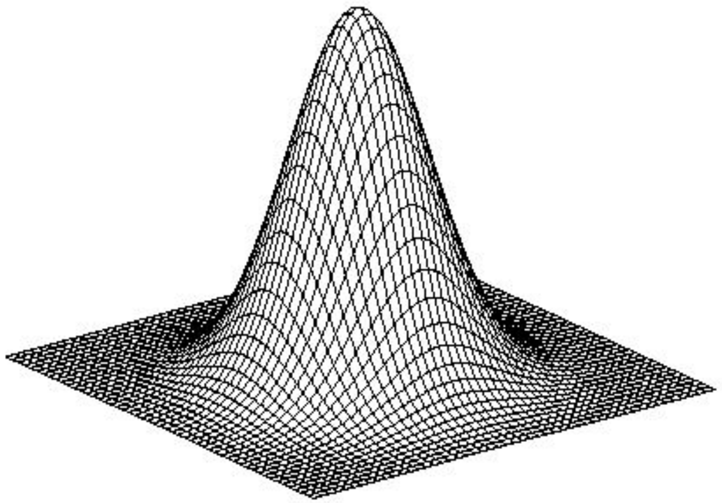
$\frac{d}{dx} g$

$f * \frac{d}{dx} g$

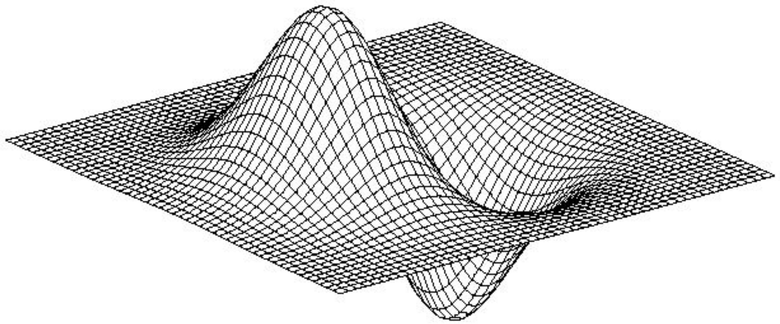


Source: S. Seitz

Derivative of Gaussian filter (central derivative)



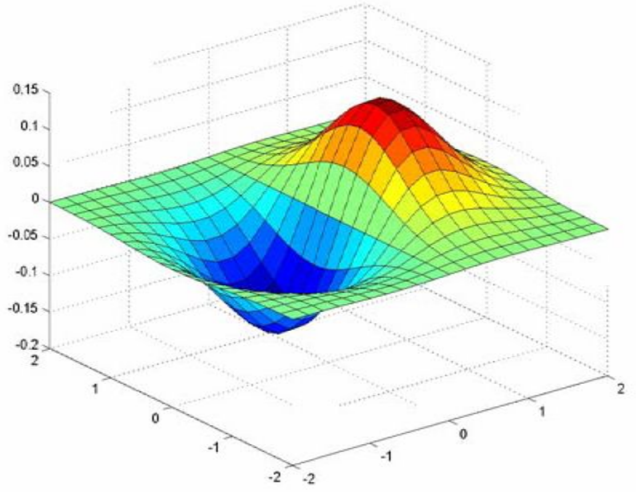
* [1 0 -1] =



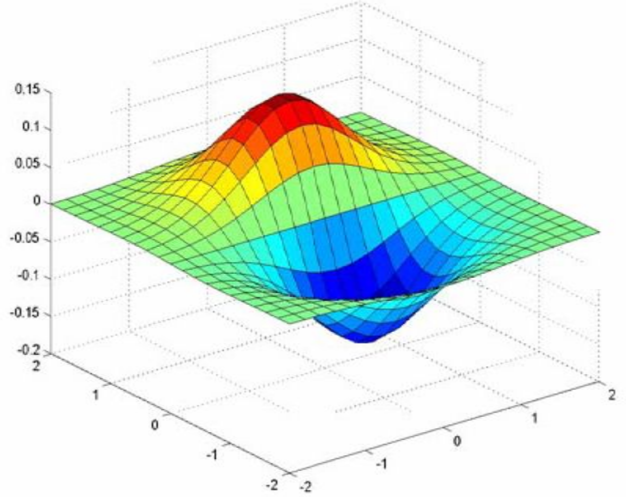
2D-gaussian

x - derivative

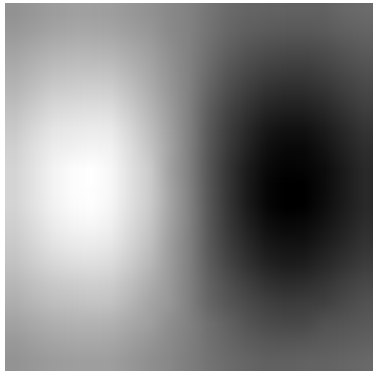
Derivative of Gaussian filter

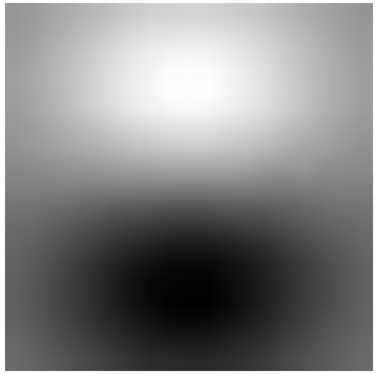


x-direction



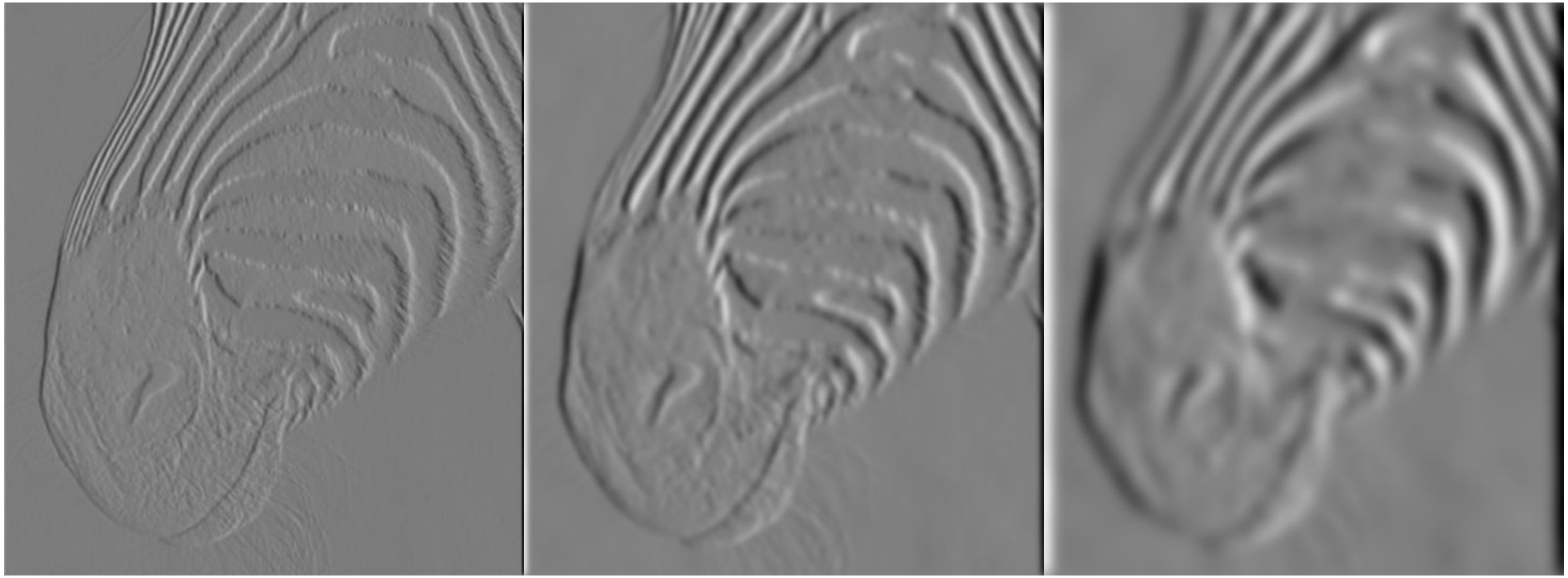
y-direction





Derivative of Gaussian filter

Tradeoff between smoothing at different scales



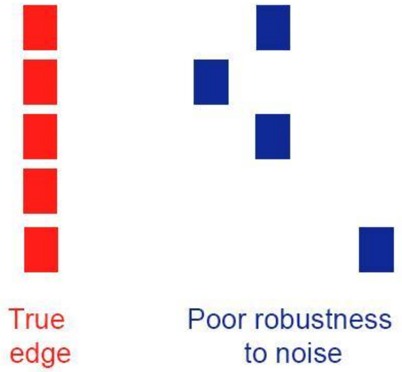
1 pixel | 3 pixels | 7 pixels

Smoothed derivative removes noise, but blurs edge.
Also finds edges at different "scales".

Source: D. Forsyth

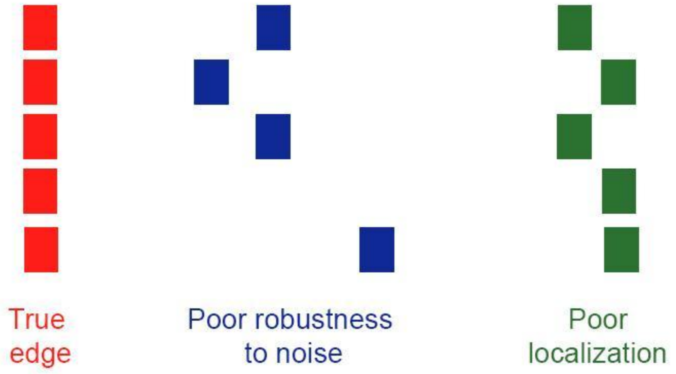
Designing an edge detector

- Criteria for an "optimal" edge detector:
  - **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)



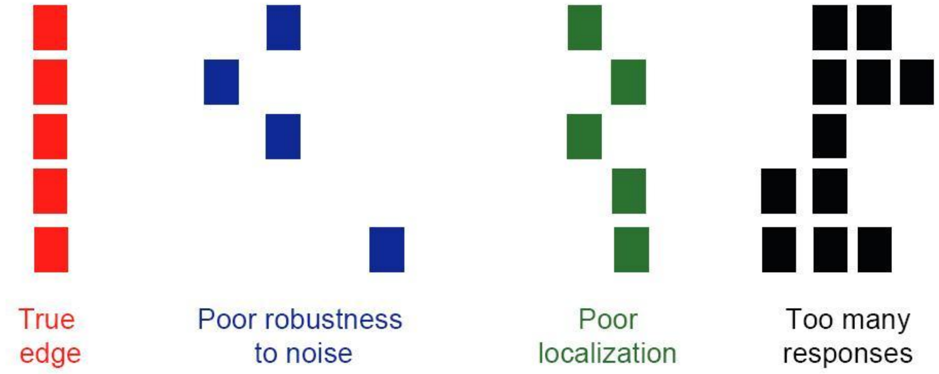
Designing an edge detector

- Criteria for an "optimal" edge detector:
  - **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)
  - **Good localization:** the edges detected must be as close as possible to the true edges



Designing an edge detector

- Criteria for an “optimal” edge detector:
  - **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges)
  - **Good localization:** the edges detected must be as close as possible to the true edges
  - **Single response:** the detector must return one point only for each true edge point; that is, minimize the number of local maxima around the true edge



Summary

- Edge detection
- Image Gradients
- A simple edge detector

Next time: Detecting lines